WhoGoesThere

Ishtiyaque Ahmad, Gwyneth Allwright, Abtin Bateni, Sabrina Tsui

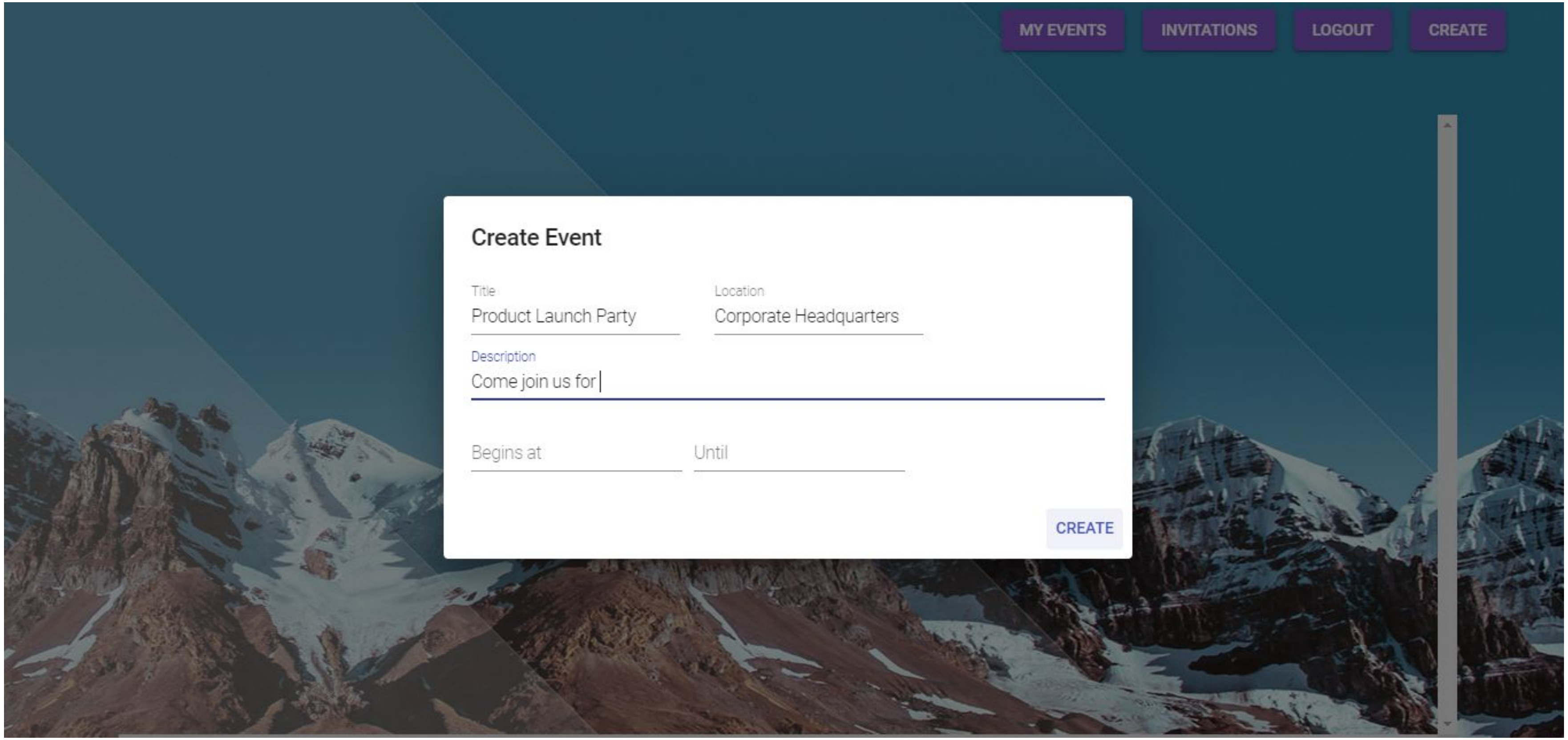
Outline

- Application Walkthrough
- Load Test Setup
- Optimizations
- Vertical Scaling Results
- Horizontal Scaling Results
- Evaluation

Events Application

- Designed for large-scale events
    - Large user base
    - Many events
- Two broad categories of users
    - Event organizers
    - Event invitees

Application Features: Creating an Event



Create Event

Title: Product Launch Party

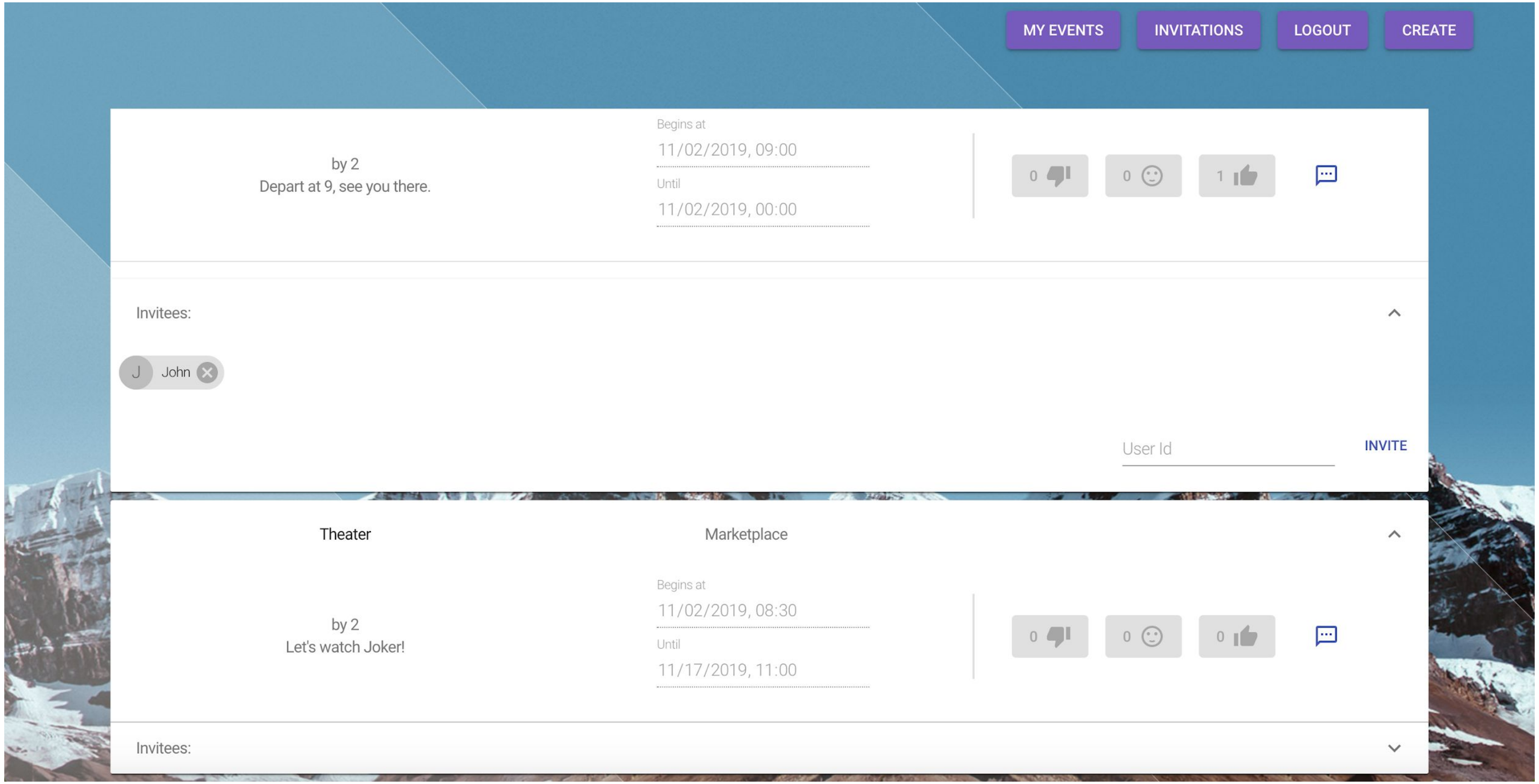
Location: Corporate Headquarters

Description: Come join us for

Begins at

Until

Application Features: My Events Page



by 2
Depart at 9, see you there.

Begins at
11/02/2019, 09:00
Until
11/02/2019, 00:00

Invitees:

Theater

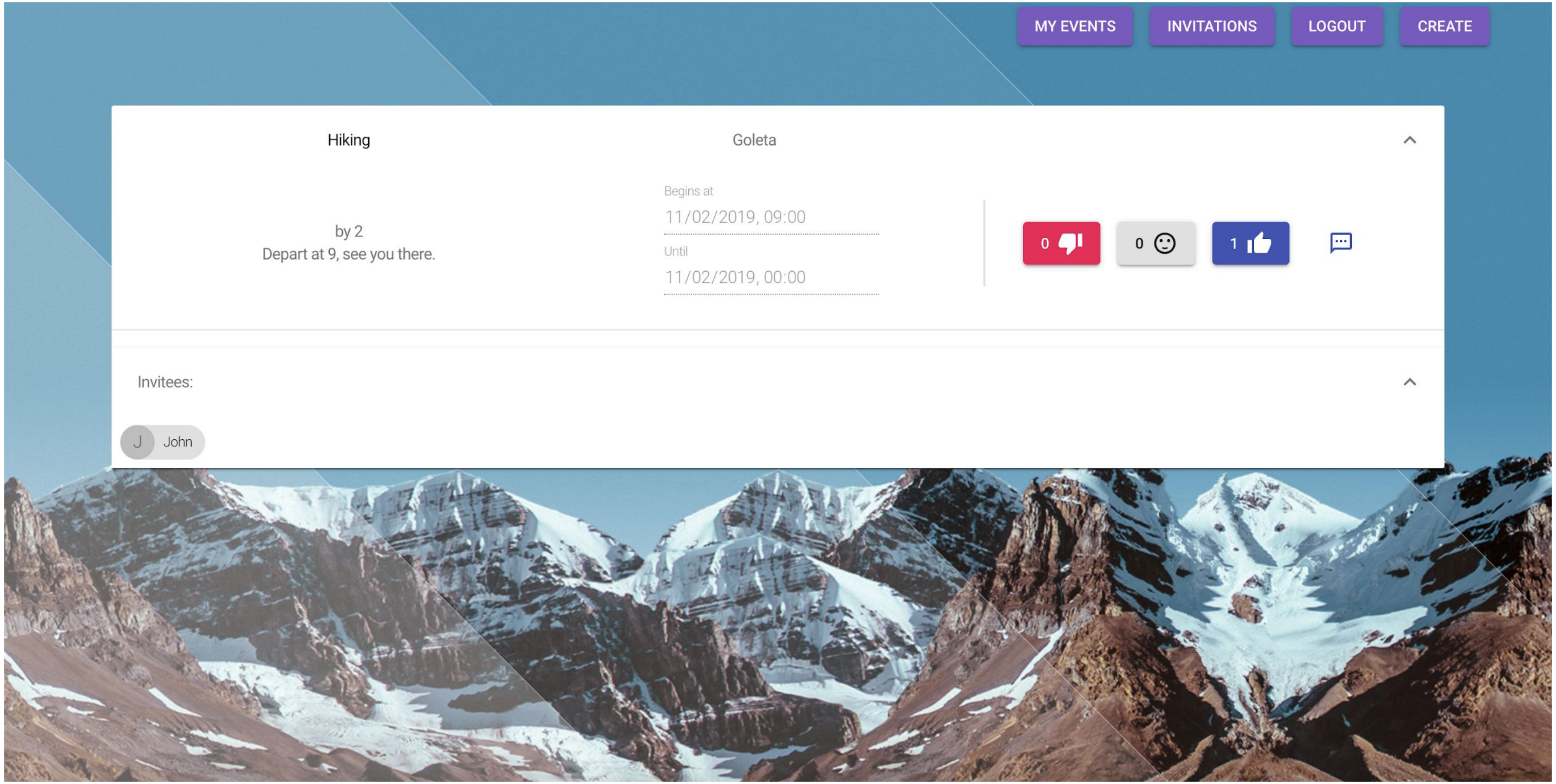
Marketplace

by 2
Let's watch Joker!

Begins at
11/02/2019, 08:30
Until
11/17/2019, 11:00

Invitees:

Application Features: Invitations Page



Hiking

Goleta

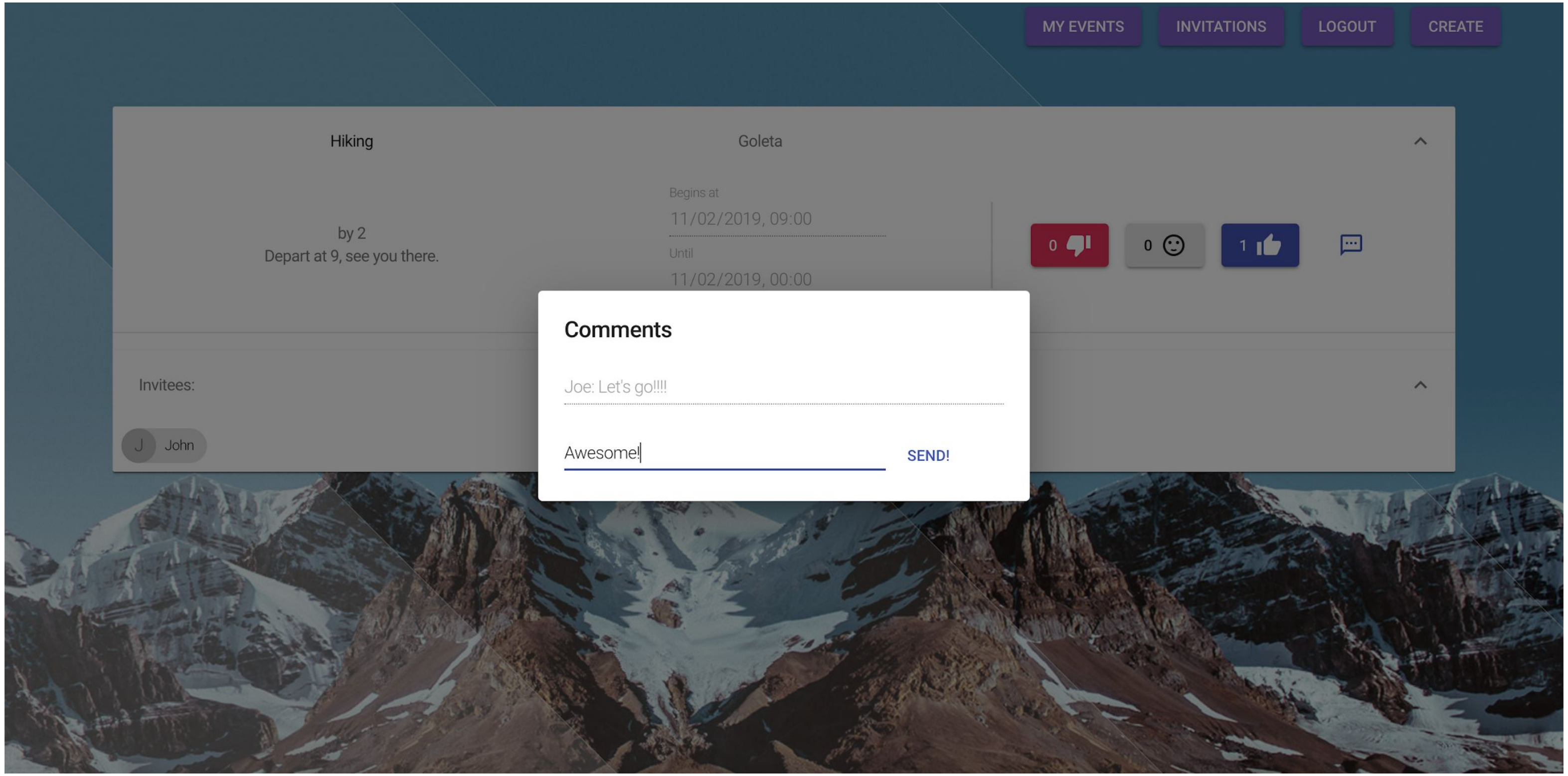
by 2
Depart at 9, see you there.

Begins at
11/02/2019, 09:00

Until
11/02/2019, 00:00

Invitees:

Application Features: Adding a Comment



Hiking

Goleta

by 2
Depart at 9, see you there.

Begins at
11/02/2019, 09:00

Until
11/02/2019, 00:00

Invitees:

J John

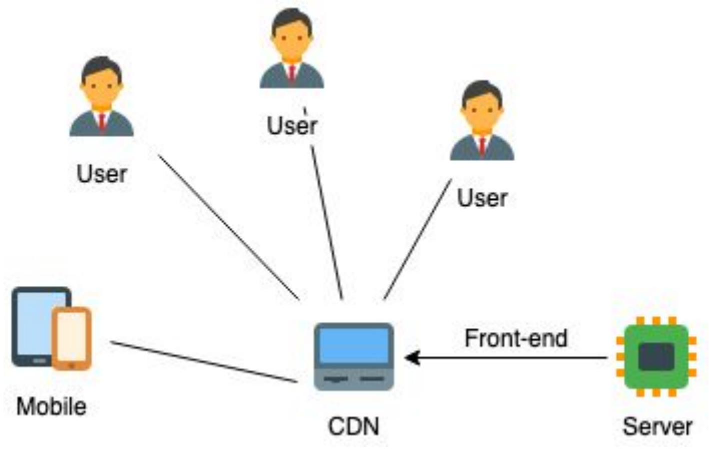
Comments

Joe: Let's go!!!!

Awesome!

Separate Client and Backend

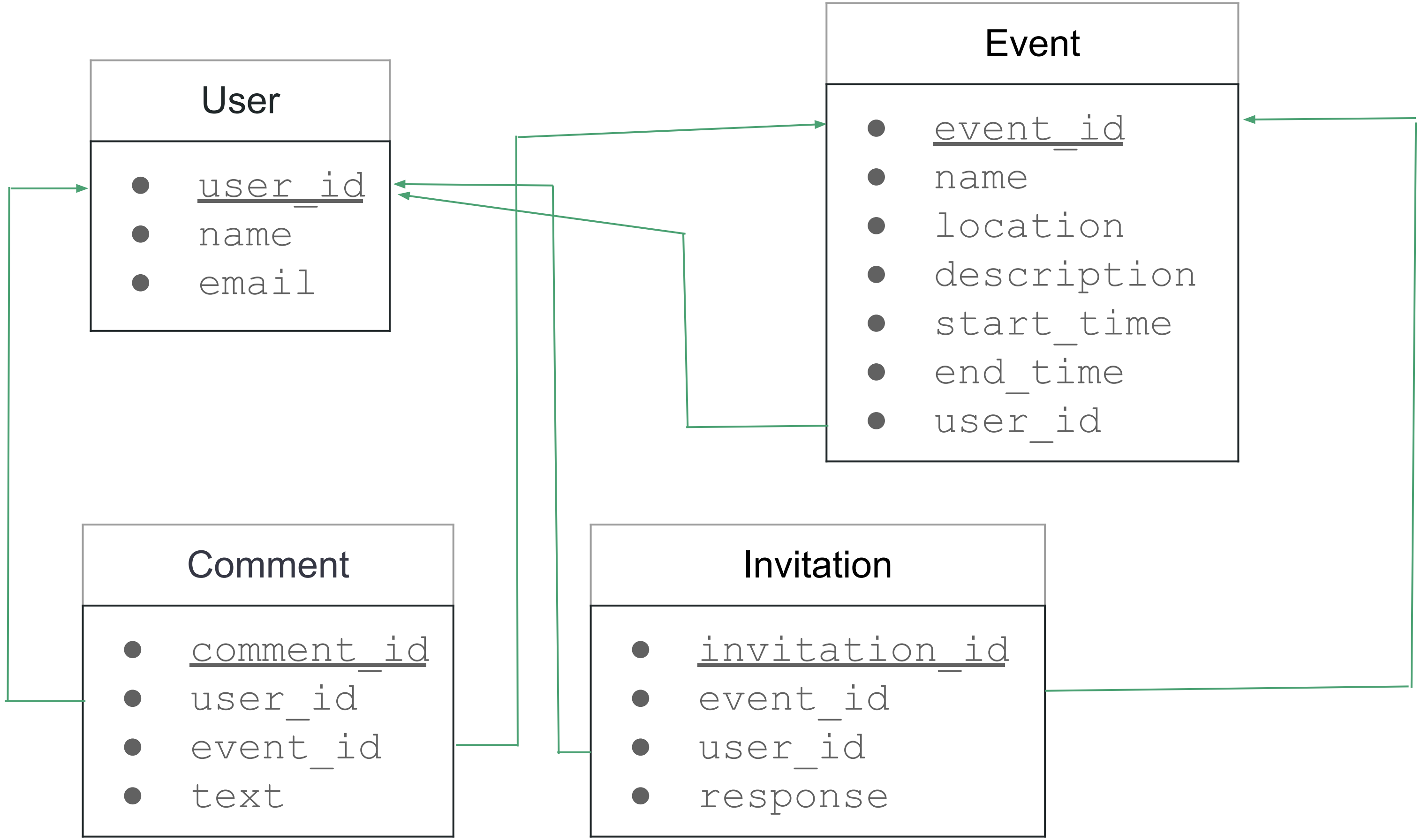
- API based service vs web pages
- Endpoints return JSON instead of html views
- Reduces code dependencies



Data Model

- Users
  - Name
  - Email
- Events
  - Name
  - Location
  - Description
  - Start time
  - End time
  - User
- Invitations
  - Event
  - User
  - Response
- Comments
  - User
  - Event
  - Text

Data Model


User
user_id
name
email
Event
event_id
name
location
description
start_time
end_time
user_id
Comment
comment_id
user_id
event_id
text
Invitation
invitation_id
event_id
user_id
response

Tsung Test Workflow

- **Organizer (50% probability)**
  - Navigate to the main page displaying all their events.
  - **Create a new event.**
  - Redirect to the events page with the new event.
  - **Invite ten other users** to the event.

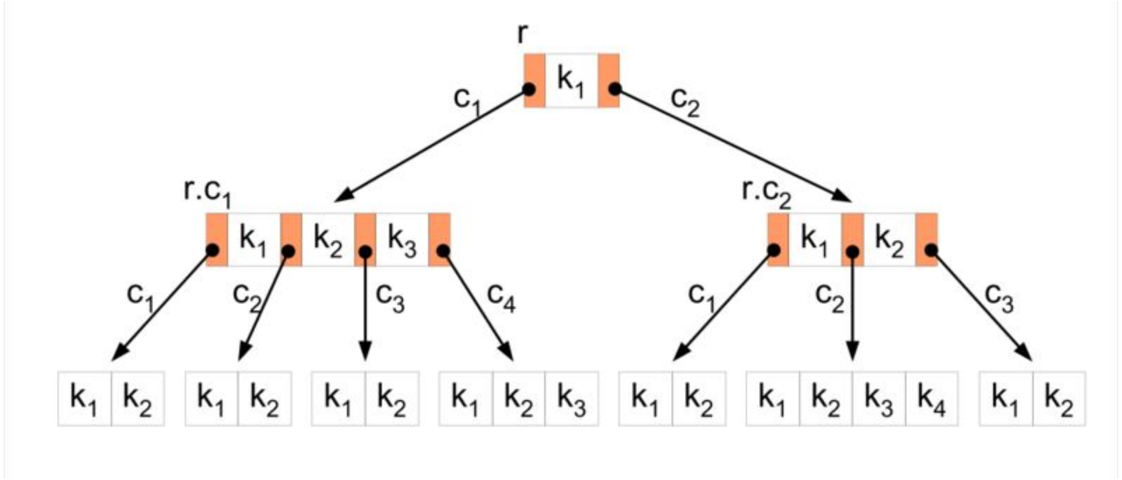
- **Invitee (50% probability)**
  - Navigate to the main page displaying all their events.
  - Navigate to the first event on the page.
  - **RSVP to the event.**
  - View the comments for the event.
  - **Post a comment** to the event.

Optimizations Outline

- Database:
    - Indexing
    - Queries (JOIN versus no JOIN)
- Application Server:
    - JBuilder
    - JSON Compression (gzip)

Database Optimizations: Indexing

- Default: data is stored in a B-tree indexed by primary key
- We added indexing on non-default keys
- Faster lookup of values



Database Optimizations: Indexing

User

- user_id
- name
- email

Event

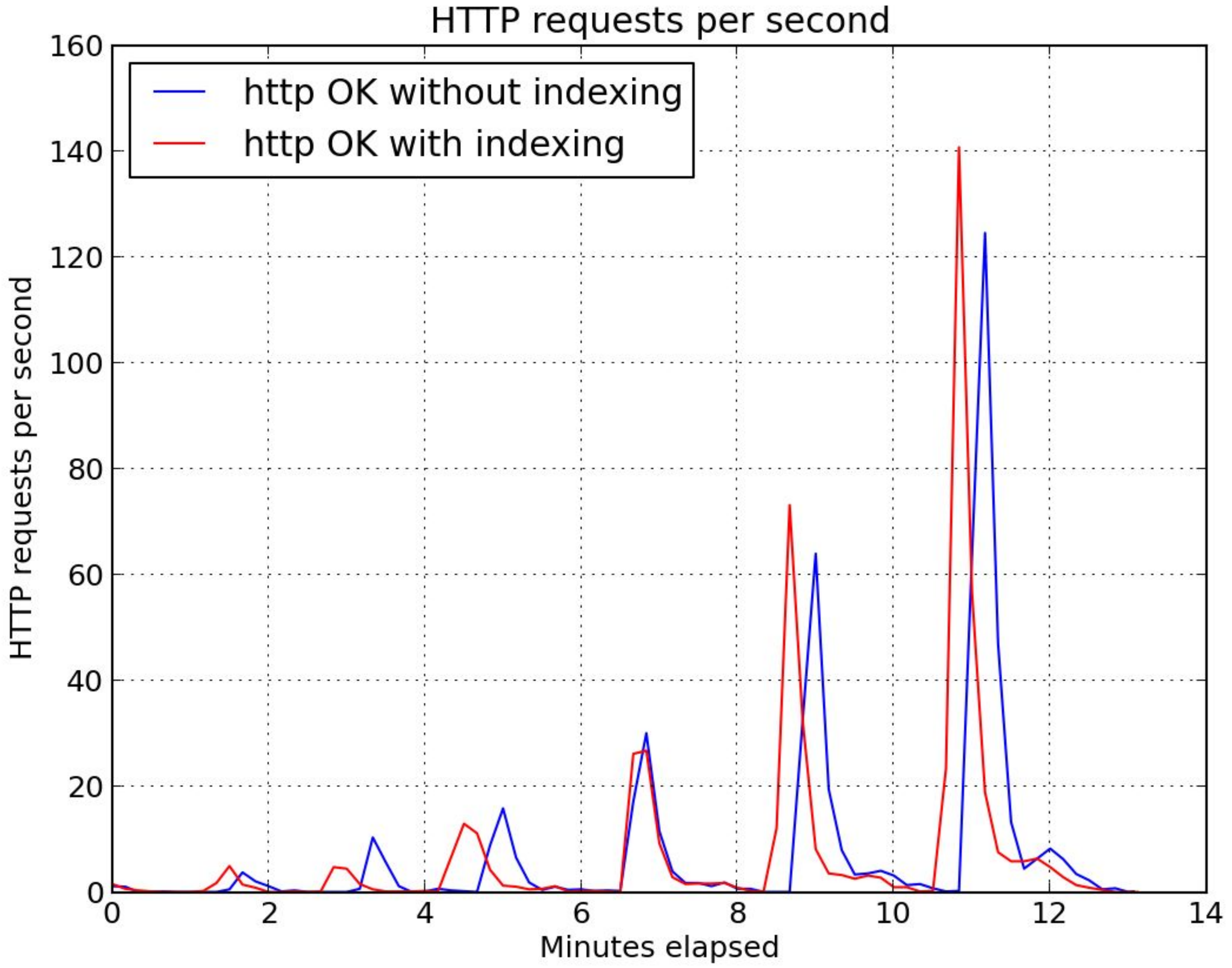
- event_id
- name
- location
- description
- start_time
- end_time
- **user_id**

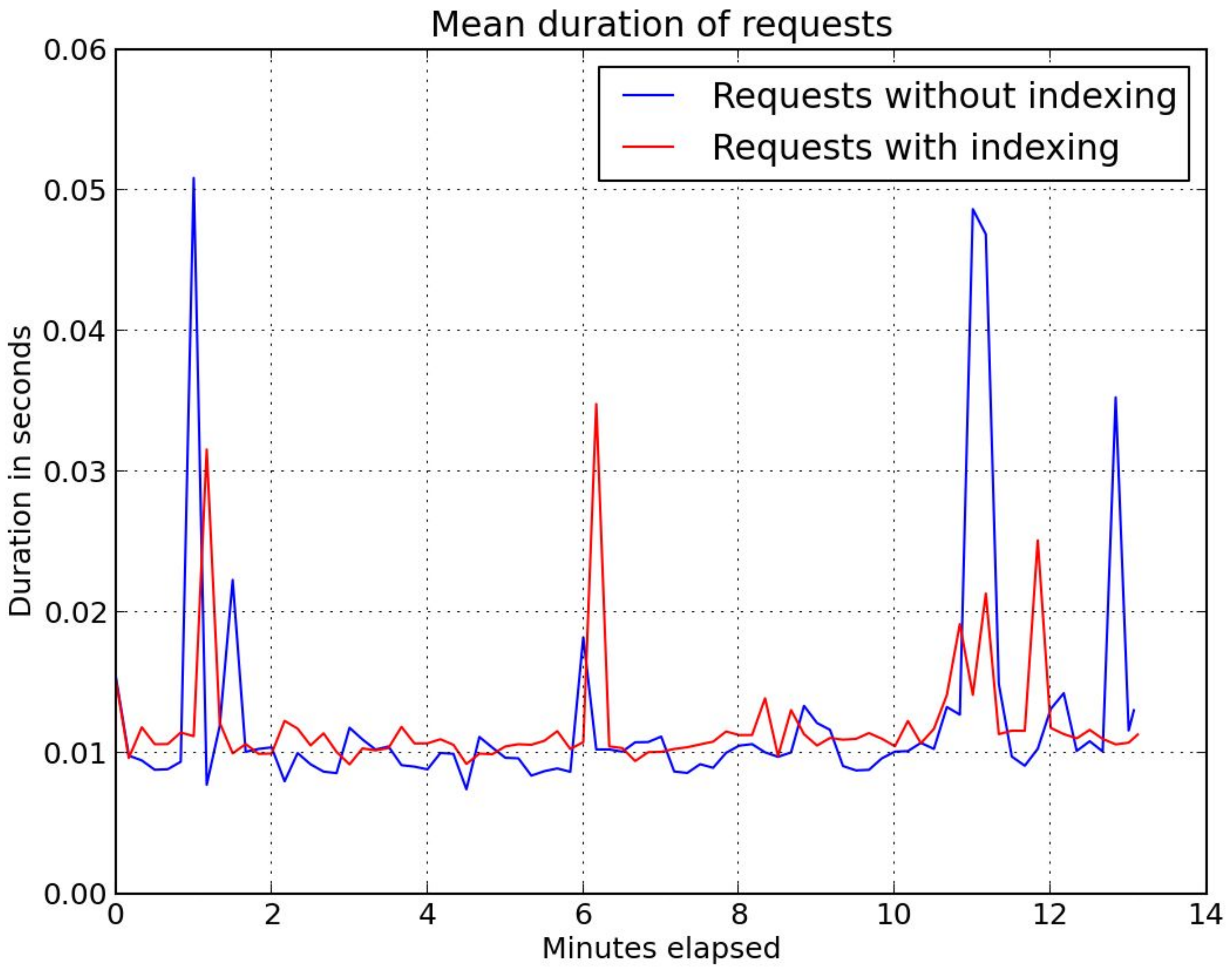
Invitation

- invitation_id
- **event_id**
- **user_id**
- response

Comment

- comment_id
- **user_id**
- **event_id**
- text

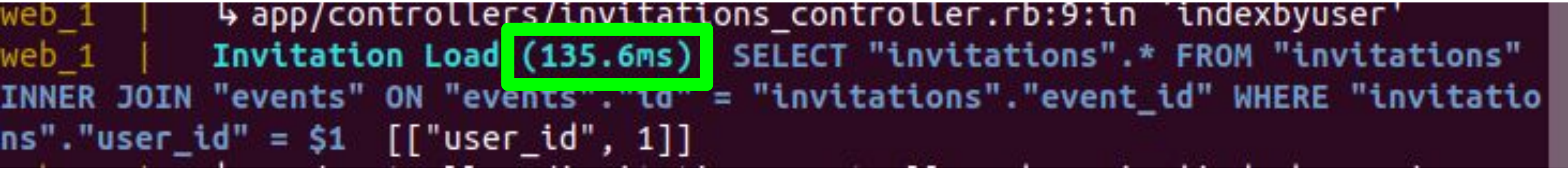

HTTP requests per second
http OK without indexing
http OK with indexing
HTTP requests per second
Minutes elapsed
0
20
40
60
80
100
120
140
160
0
2
4
6
8
10
12
14


Mean duration of requests
Requests without indexing
Requests with indexing
Duration in seconds
Minutes elapsed
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0
2
4
6
8
10
12
14

Optimizations Outline

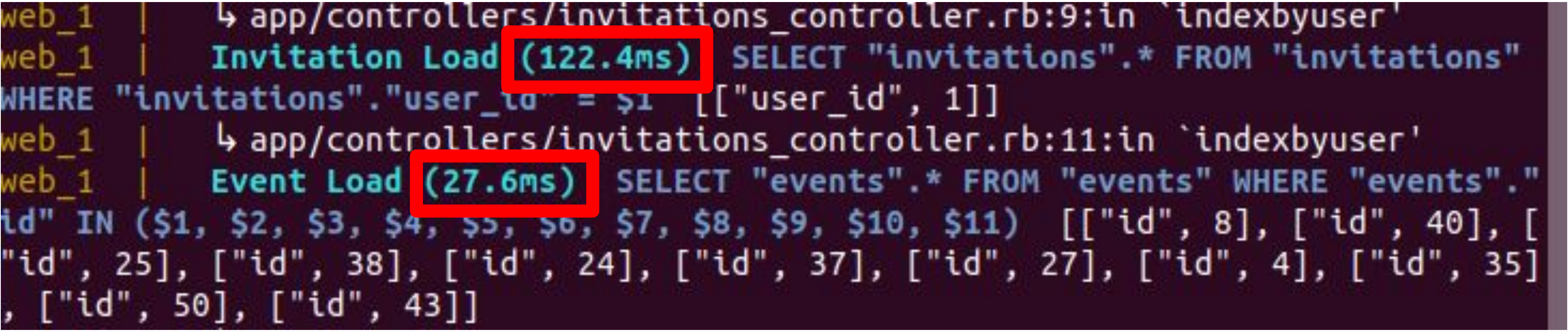
- Database:
    - Indexing
    - Queries (JOIN versus no JOIN)
- Application Server:
    - JBuilder
    - JSON Compression (gzip)

JOIN versus no JOIN



```
web_1  |    ↳ app/controllers/invitations_controller.rb:9:in `indexbyuser'
web_1  |    Invitation Load (135.6ms)  SELECT "invitations".* FROM "invitations"
INNER JOIN "events" ON "events"."id" = "invitations"."event_id" WHERE "invitatio
ns"."user_id" = $1  [["user_id", 1]]
```

Performance with join



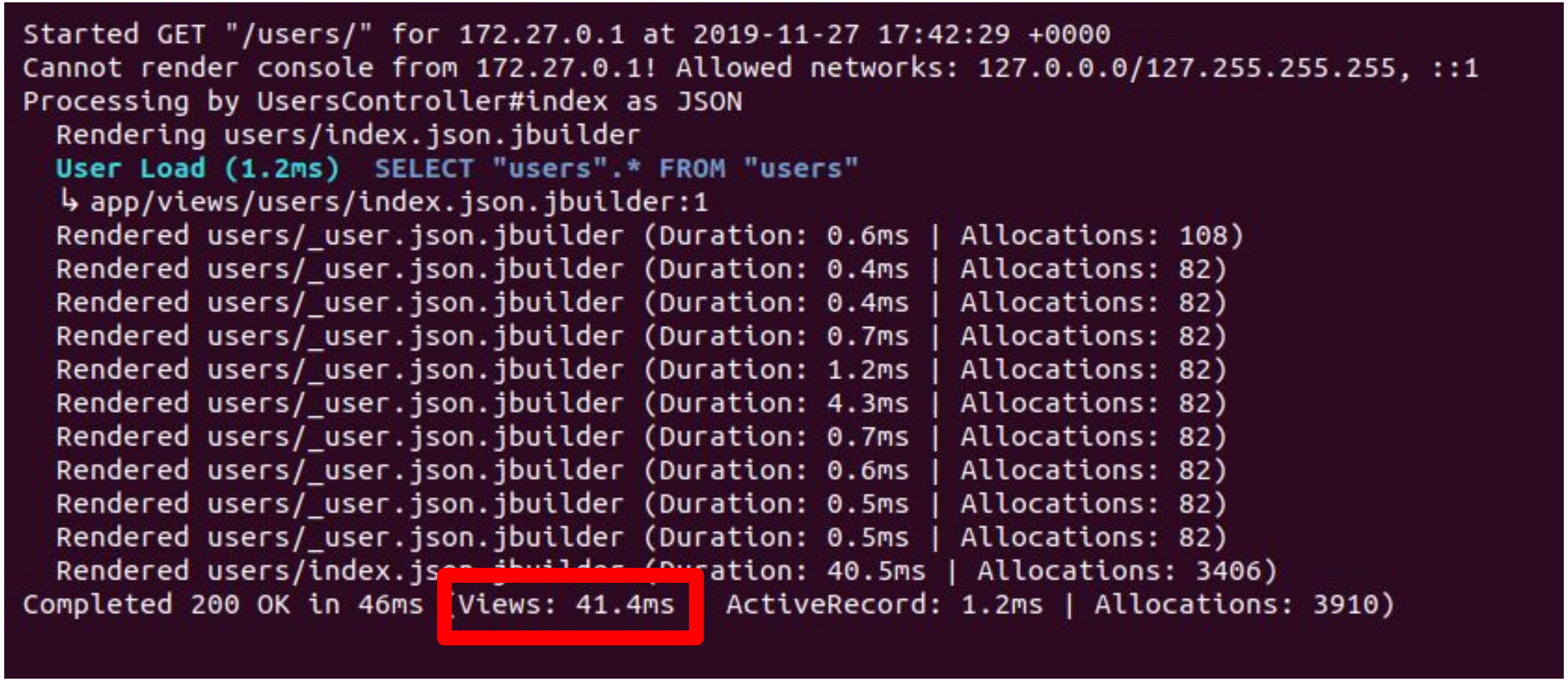
```
web_1  |    ↳ app/controllers/invitations_controller.rb:9:in `indexbyuser'
web_1  |    Invitation Load (122.4ms)  SELECT "invitations".* FROM "invitations"
WHERE "invitations"."user_id" = $1  [["user_id", 1]]
web_1  |    ↳ app/controllers/invitations_controller.rb:11:in `indexbyuser'
web_1  |    Event Load (27.6ms)  SELECT "events".* FROM "events" WHERE "events"."
id" IN ($1, $2, $3, $4, $5, $6, $7, $8, $9, $10, $11)  [["id", 8], ["id", 40], [
"id", 25], ["id", 38], ["id", 24], ["id", 37], ["id", 27], ["id", 4], ["id", 35]
, ["id", 50], ["id", 43]]
```

Performance without join

Optimizations Outline

- Database:
  - Indexing
  - Queries (JOIN versus no JOIN)
- Application Server:
  - JBuilder
  - JSON Compression (gzip)

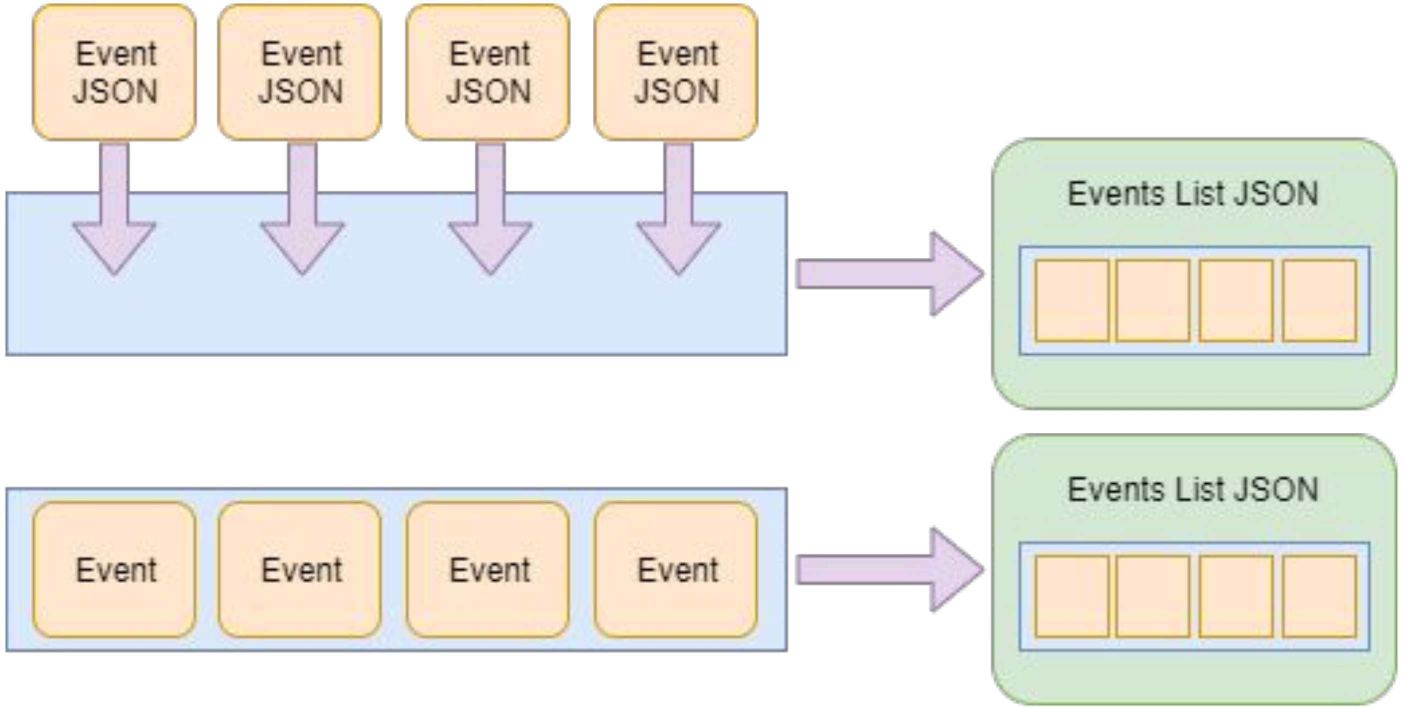
JSON Optimization



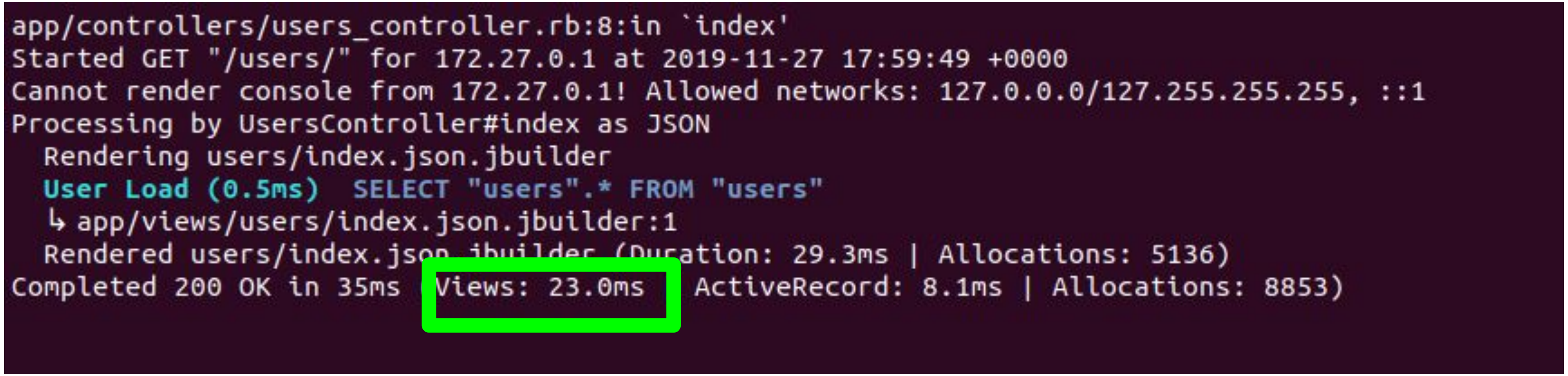
```
Started GET "/users/" for 172.27.0.1 at 2019-11-27 17:42:29 +0000
Cannot render console from 172.27.0.1! Allowed networks: 127.0.0.0/127.255.255.255, ::1
Processing by UsersController#index as JSON
  Rendering users/index.json.jbuilder
  User Load (1.2ms)  SELECT "users".* FROM "users"
  ↳ app/views/users/index.json.jbuilder:1
  Rendered users/_user.json.jbuilder (Duration: 0.6ms | Allocations: 108)
  Rendered users/_user.json.jbuilder (Duration: 0.4ms | Allocations: 82)
  Rendered users/_user.json.jbuilder (Duration: 0.4ms | Allocations: 82)
  Rendered users/_user.json.jbuilder (Duration: 0.7ms | Allocations: 82)
  Rendered users/_user.json.jbuilder (Duration: 1.2ms | Allocations: 82)
  Rendered users/_user.json.jbuilder (Duration: 4.3ms | Allocations: 82)
  Rendered users/_user.json.jbuilder (Duration: 0.7ms | Allocations: 82)
  Rendered users/_user.json.jbuilder (Duration: 0.6ms | Allocations: 82)
  Rendered users/_user.json.jbuilder (Duration: 0.5ms | Allocations: 82)
  Rendered users/_user.json.jbuilder (Duration: 0.5ms | Allocations: 82)
  Rendered users/index.json.jbuilder (Duration: 40.5ms | Allocations: 3406)
Completed 200 OK in 46ms (Views: 41.4ms | ActiveRecord: 1.2ms | Allocations: 3910)
```

JSON Optimization

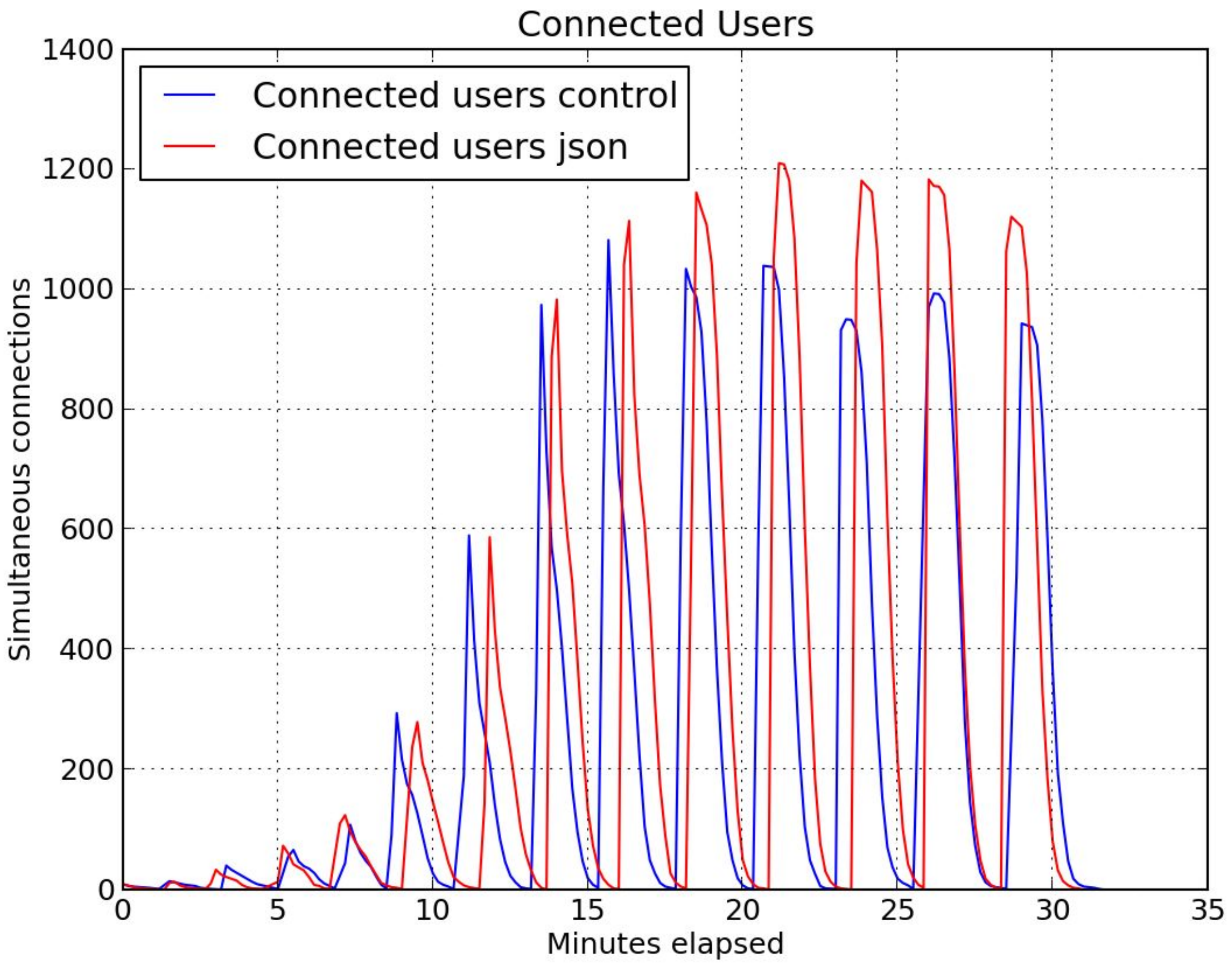
- JBuilder Optimization in Rails



JSON Optimization



```
app/controllers/users_controller.rb:8:in `index'
Started GET "/users/" for 172.27.0.1 at 2019-11-27 17:59:49 +0000
Cannot render console from 172.27.0.1! Allowed networks: 127.0.0.0/127.255.255.255, ::1
Processing by UsersController#index as JSON
  Rendering users/index.json.jbuilder
  User Load (0.5ms)  SELECT "users".* FROM "users"
  ↳ app/views/users/index.json.jbuilder:1
  Rendered users/index.json.jbuilder (Duration: 29.3ms | Allocations: 5136)
Completed 200 OK in 35ms (Views: 23.0ms | ActiveRecord: 8.1ms | Allocations: 8853)
```


Connected Users
Connected users control
Connected users json
Simultaneous connections
Minutes elapsed
0
200
400
600
800
1000
1200
1400
0
5
10
15
20
25
30
35

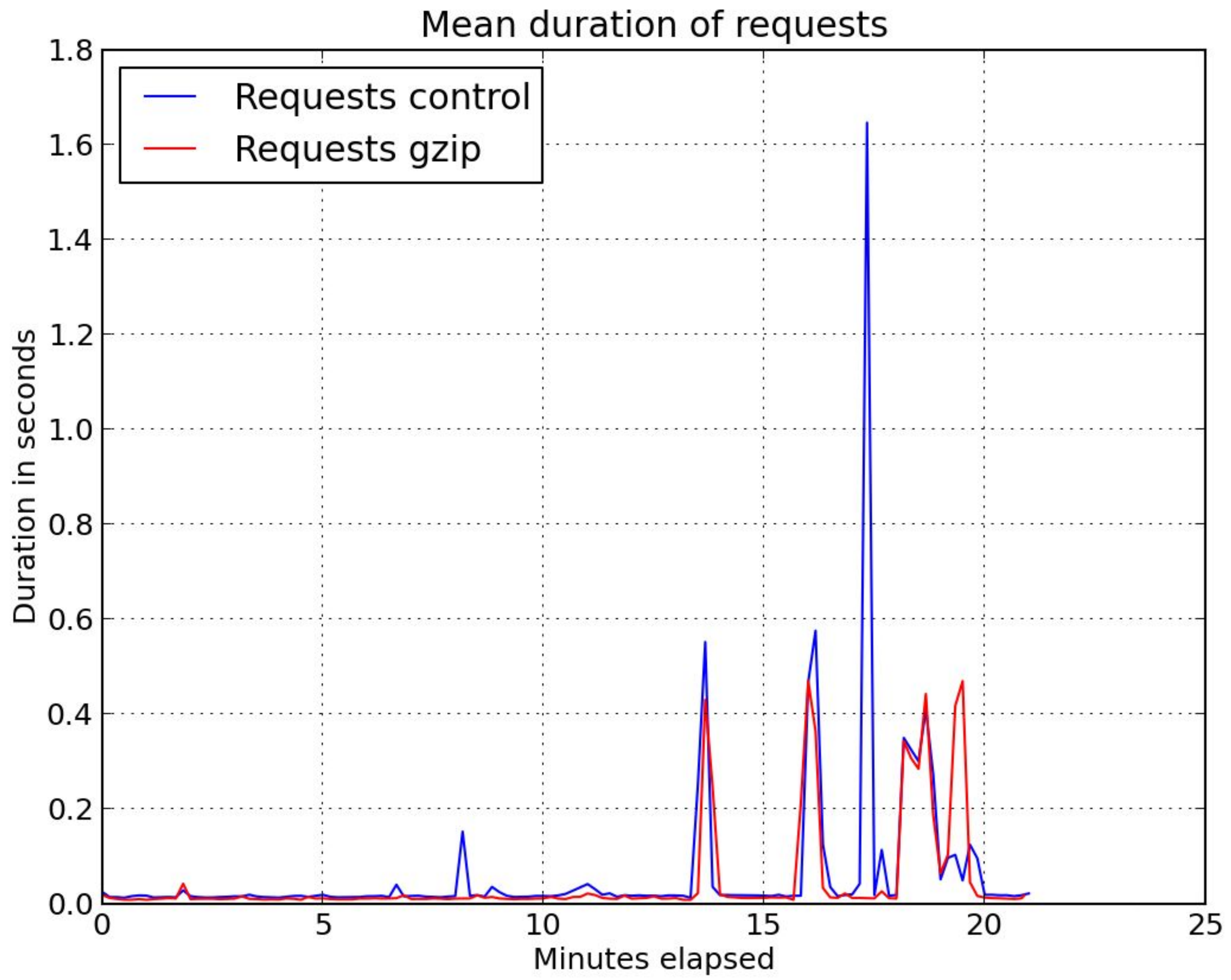
Optimizations Outline

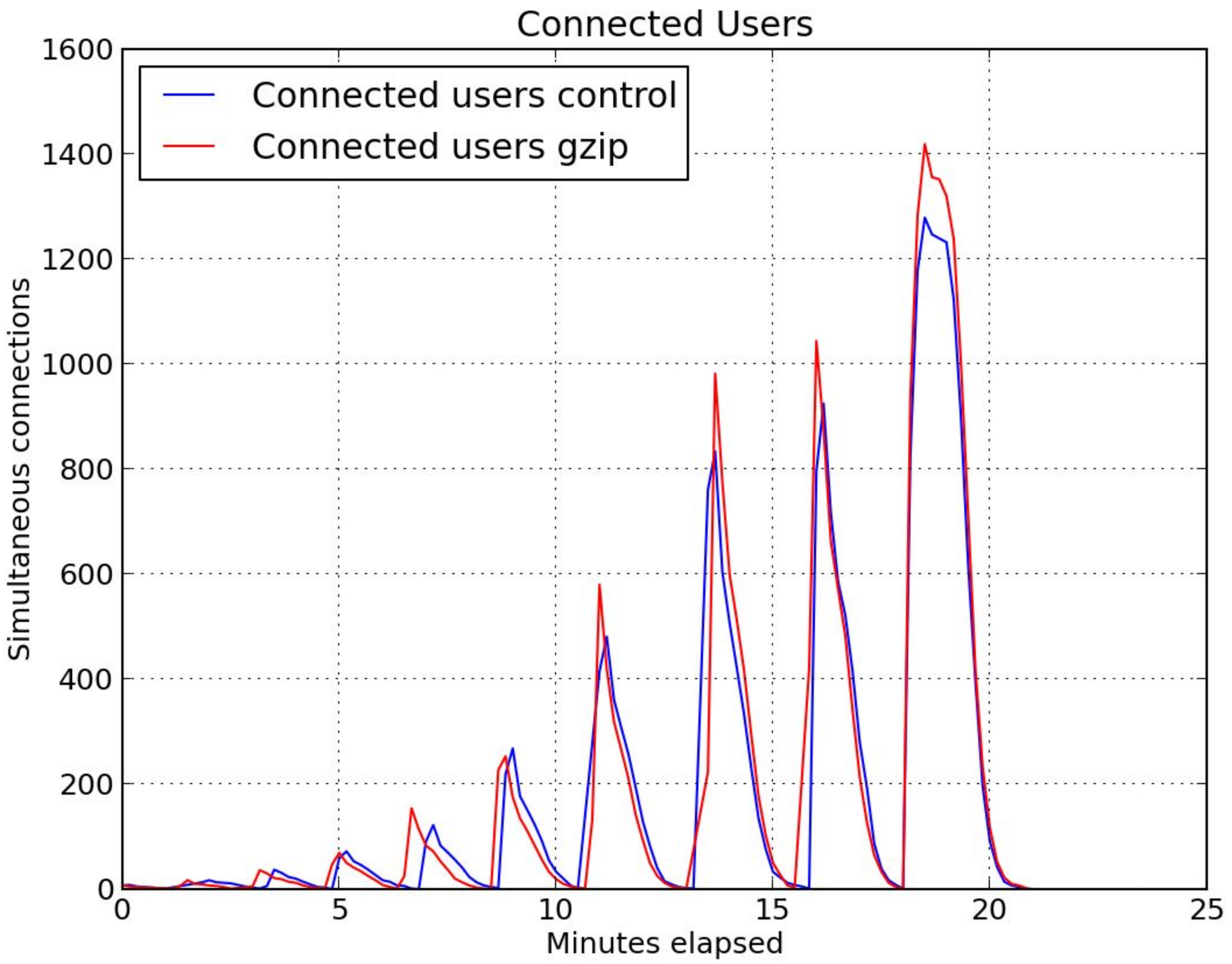
- Database:
  - Indexing
  - Queries (JOIN versus no JOIN)
- Application Server:
  - JBuilder
  - JSON Compression (gzip)

JSON Compression

```
GET /encrypted-area HTTP/1.1
Host: www.example.com
Accept-Encoding: gzip, deflate
```

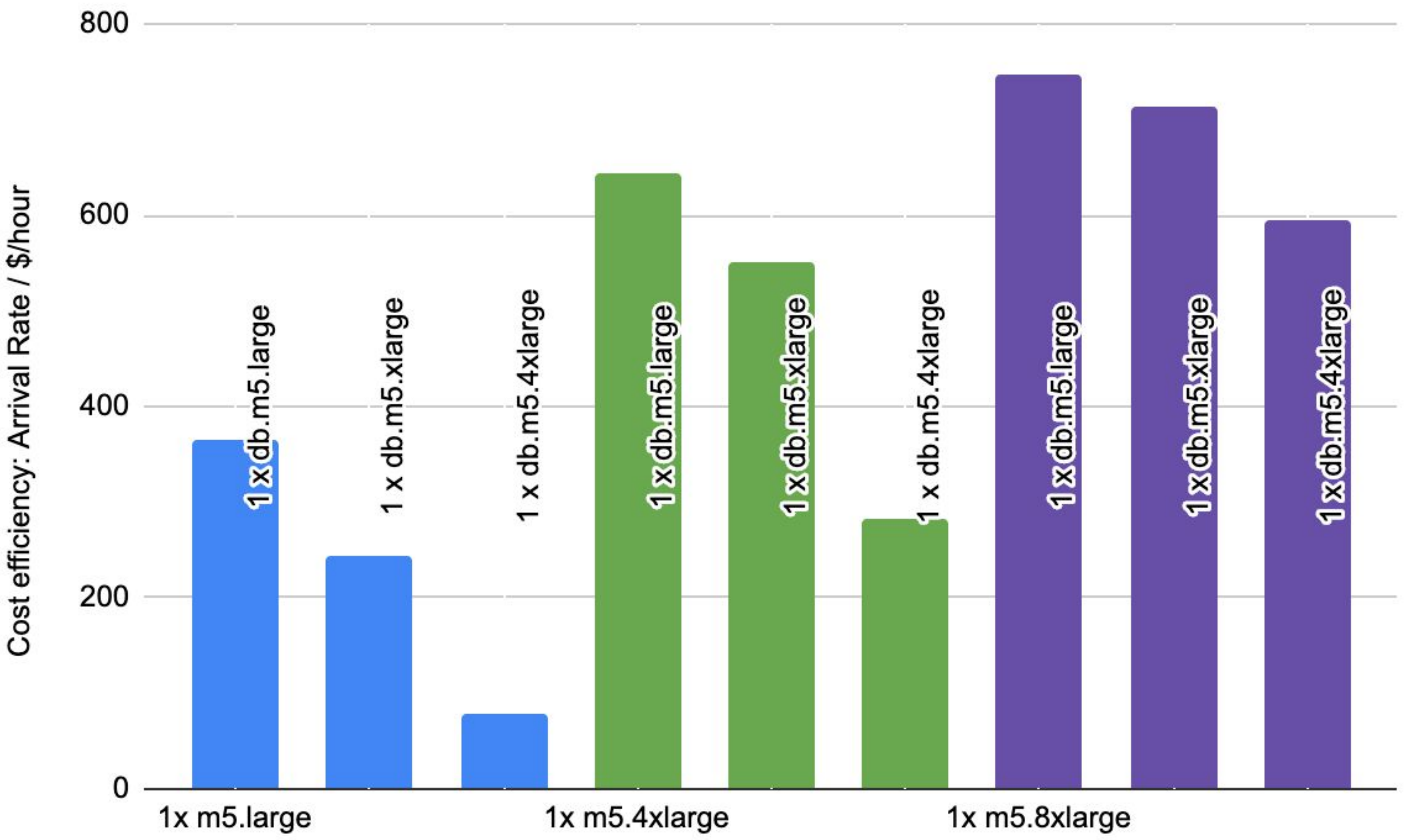
```
HTTP/1.1 200 OK
Date: mon, 26 June 2016 22:38:34 GMT
Server: Apache/1.3.3.7 (Unix)  (Red-Hat/Linux)
Last-Modified: Wed, 08 Jan 2003 23:11:55 GMT
Accept-Ranges: bytes
Content-Length: 438
Connection: close
Content-Type: text/html; charset=UTF-8
Content-Encoding: gzip
```


Mean duration of requests
Requests control
Requests gzip
Duration in seconds
Minutes elapsed
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
1.8
0
5
10
15
20
25


Connected Users
Connected users control
Connected users gzip
Simultaneous connections
Minutes elapsed
0
200
400
600
800
1000
1200
1400
1600
0
5
10
15
20
25

Vertical Scaling Results

Application Instance	Database Instance	Max Arrival Rate Before 5XX Error (users/second)
1x m5.large	1 x db.m5.large	100
	1 x db.m5.xlarge	110
	1 x db.m5.4xlarge	120
1x m5.4xlarge	1 x db.m5.large	610
	1 x db.m5.xlarge	620
	1 x db.m5.4xlarge	620
1x m5.8xlarge	1 x db.m5.large	1280
	1 x db.m5.xlarge	1350
	1 x db.m5.4xlarge	1760

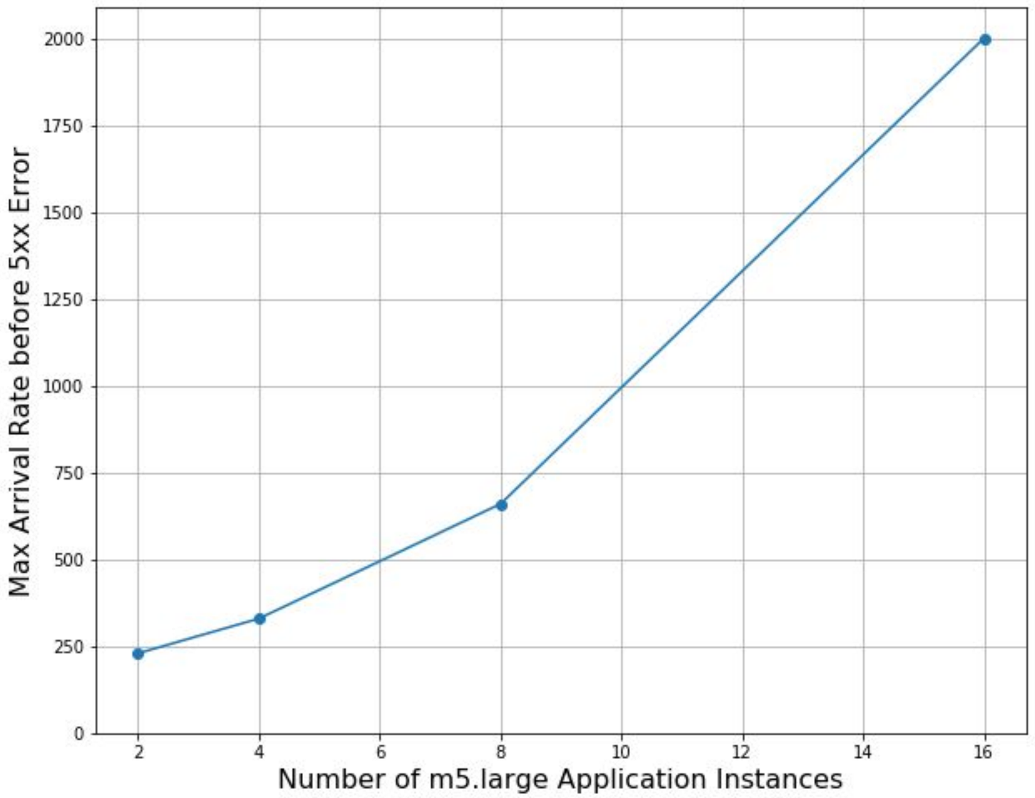

Cost efficiency: Arrival Rate / $/hour
800
600
400
200
0
1 x db.m5.large
1 x db.m5.xlarge
1 x db.m5.4xlarge
1 x db.m5.large
1 x db.m5.xlarge
1 x db.m5.4xlarge
1 x db.m5.large
1 x db.m5.xlarge
1 x db.m5.4xlarge
1x m5.large
1x m5.4xlarge
1x m5.8xlarge

Horizontal Scaling Results

Number of m5.large Application Instances	Max Arrival Rate Before 5XX Error	Max Users/s per Dollar
2	230	622
4	330	587
8	660	698
16	>2000	>1167

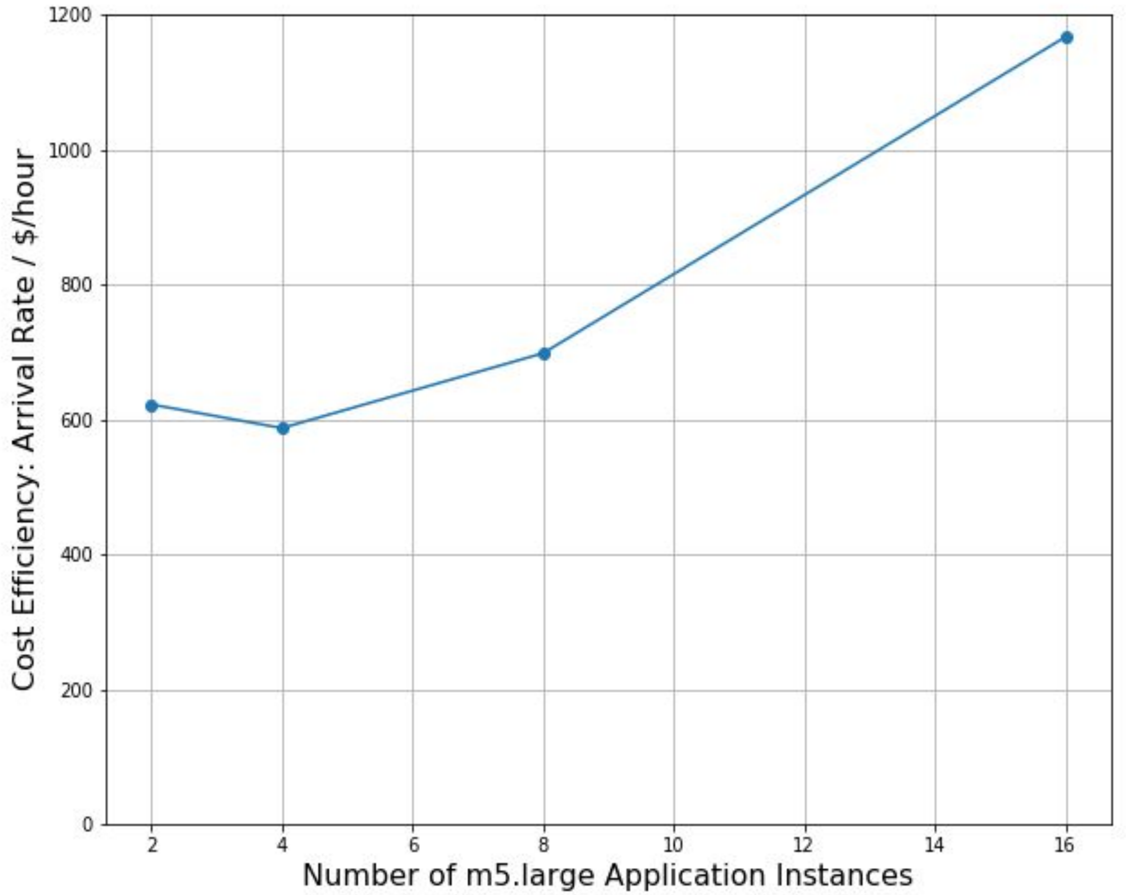
Horizontal Scaling Results

- Impact on traffic



Horizontal Scaling Results

- Impact on cost



Evaluation

- Combining all optimizations and scaling the final application
- Baseline performance
    - m5.large instance
    - Max 90 users/second tsung arrival rate
- Optimized performance
    - Adding optimizations: improvement of 10 users/second over baseline
    - Best performance: 16 x m5.large instance + db.m5.large
        - Max >2000 users tsung arrival rate

Thank You

References

- https://en.wikipedia.org/wiki/HTTP_compression
- https://commons.wikimedia.org/wiki/File:B-tree-definition.png