

# Mongo-Craft

Riaz, Brian and Nitesh

# Mongo-Craft

Riaz, Brian and Nitesh

<http://0a131ef4.ngrok.io/index.html>



mongoDB

Scalable

Object store

Flexible Schemas

Non Relational

# Schema Design

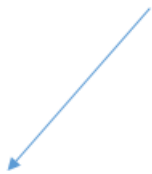
## World

A collection of Screens



## Screen

{X, Y, users, tiles}



## User

{User\_id, x, y, color,  
player information}



## Tile

{x, y, type}

## Users

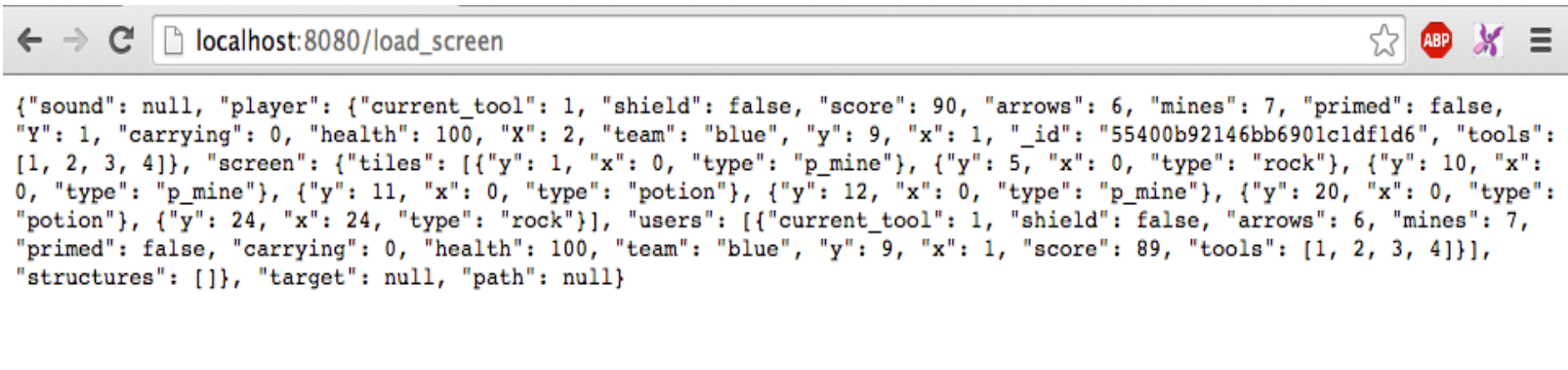
A collection of users



## User

{User\_id, X,Y, x, y, color,  
player information}

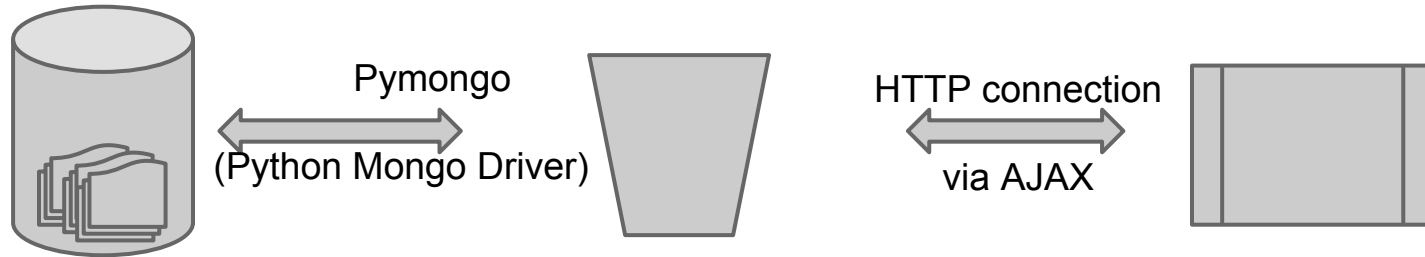
# Sending JSON over to update Screen



The image shows a browser window with the address bar containing 'localhost:8080/load\_screen'. The main content area displays a JSON object representing a game state. The JSON includes fields for 'sound', 'player', 'screen', 'users', 'structures', 'target', and 'path'. The 'player' and 'users' objects contain detailed attributes like 'current\_tool', 'shield', 'score', 'arrows', 'mines', 'primed', 'carrying', 'health', 'team', 'y', 'x', and '\_id'. The 'screen' object contains a 'tiles' array with objects for 'p\_mine', 'rock', and 'potion' at various coordinates. The 'users' array contains a single user object with a score of 89.

```
{
  "sound": null,
  "player": {
    "current_tool": 1,
    "shield": false,
    "score": 90,
    "arrows": 6,
    "mines": 7,
    "primed": false,
    "Y": 1,
    "carrying": 0,
    "health": 100,
    "X": 2,
    "team": "blue",
    "y": 9,
    "x": 1,
    "_id": "55400b92146bb6901c1df1d6",
    "tools": [1, 2, 3, 4]
  },
  "screen": {
    "tiles": [
      { "y": 1, "x": 0, "type": "p_mine" },
      { "y": 5, "x": 0, "type": "rock" },
      { "y": 10, "x": 0, "type": "p_mine" },
      { "y": 11, "x": 0, "type": "potion" },
      { "y": 12, "x": 0, "type": "p_mine" },
      { "y": 20, "x": 0, "type": "potion" },
      { "y": 24, "x": 24, "type": "rock" }
    ]
  },
  "users": [
    {
      "current_tool": 1,
      "shield": false,
      "arrows": 6,
      "mines": 7,
      "primed": false,
      "carrying": 0,
      "health": 100,
      "team": "blue",
      "y": 9,
      "x": 1,
      "score": 89,
      "tools": [1, 2, 3, 4]
    }
  ],
  "structures": [],
  "target": null,
  "path": null
}
```

# Model/Controller/View

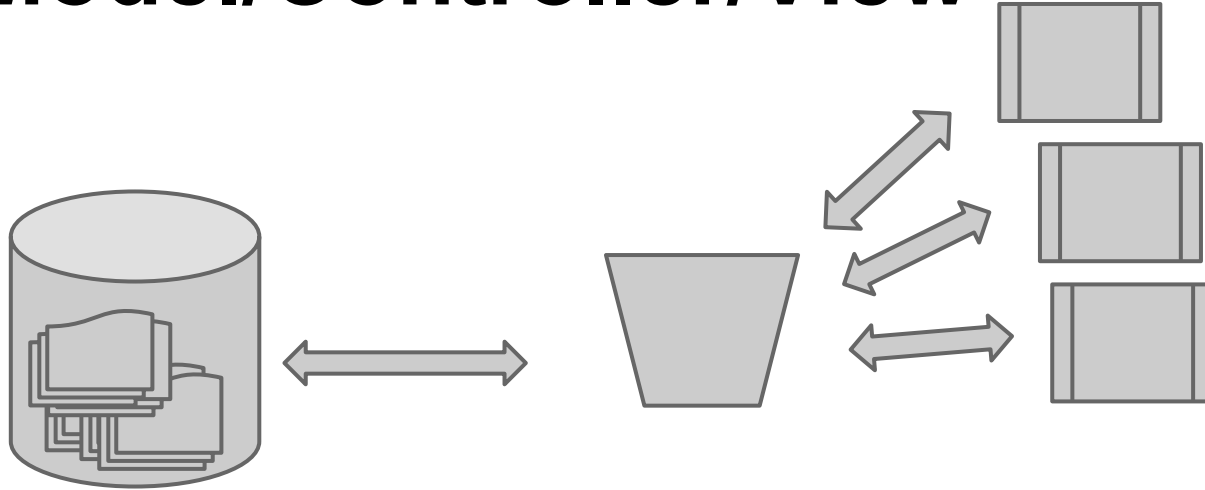


Mongo Database  
Server

Python Webserver  
using Bottle

Web-Browser  
Http Client

# Model/Controller/View

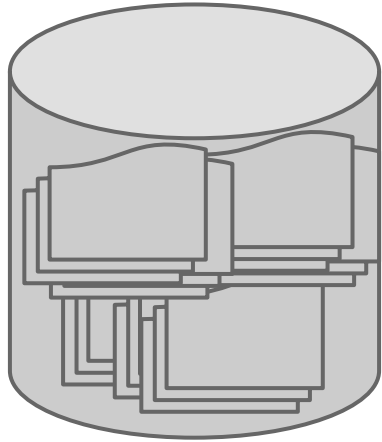


Mongo Database  
Server

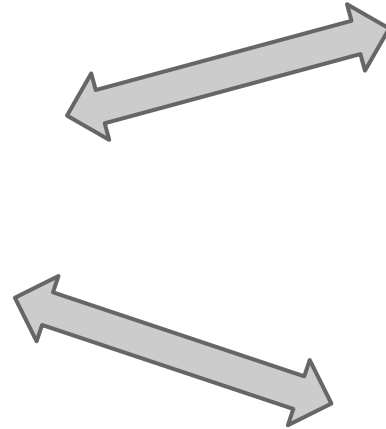
Python Webserver  
using Bottle

Multiple  
Web-Browser  
Http Clients

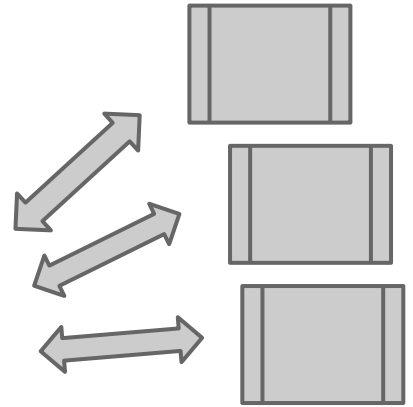
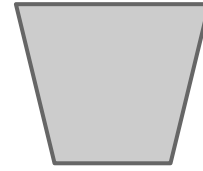
# Model/Controller/View



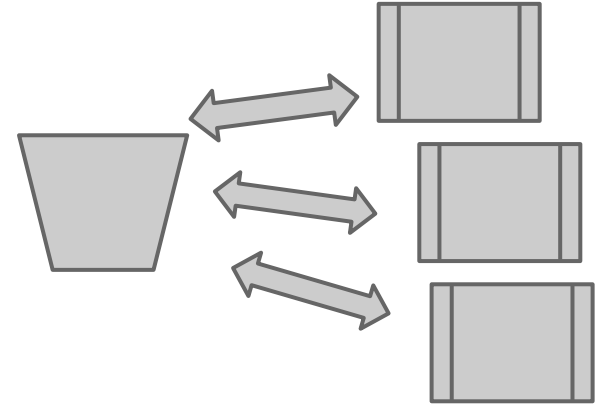
Mongo Database Server



Multiple Python Webservers

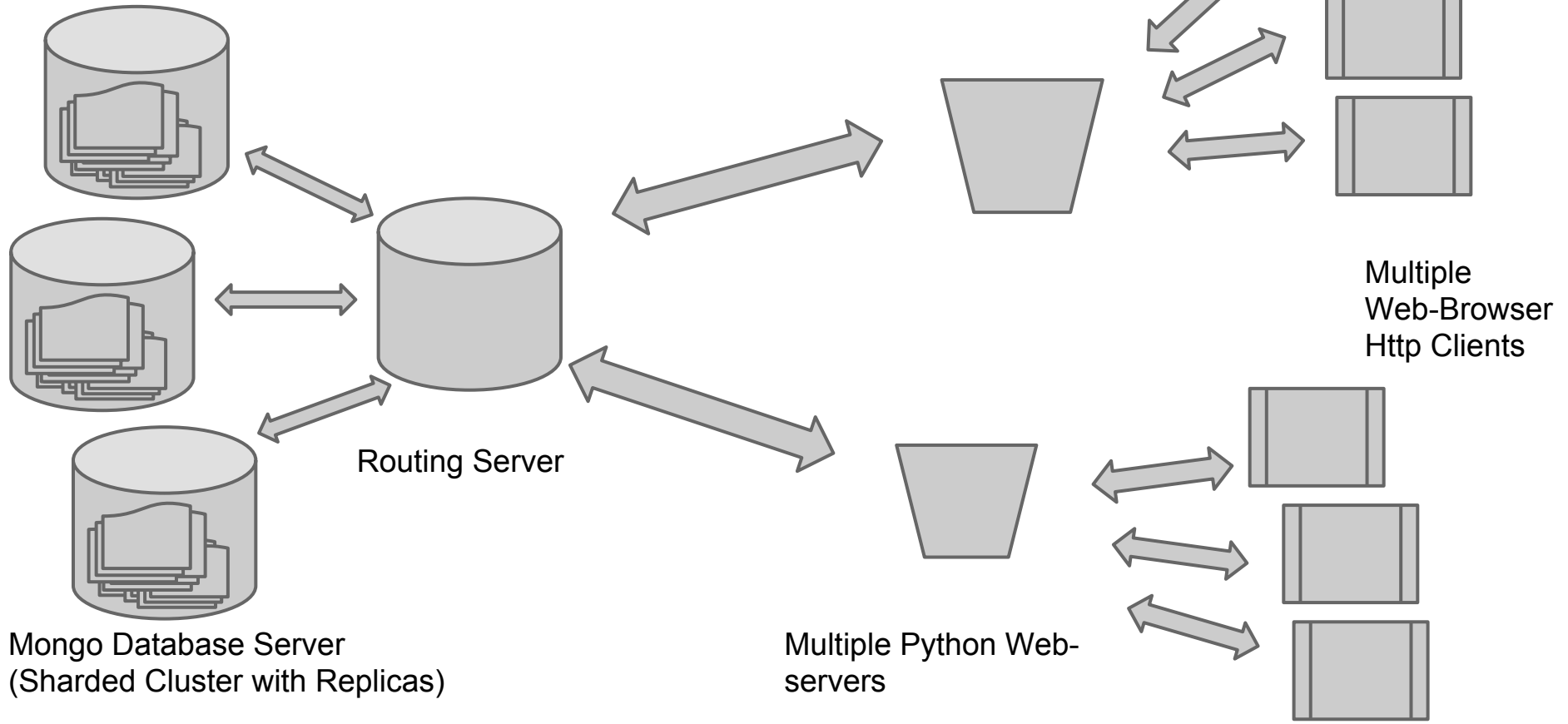


Multiple Web-Browser Http Clients





# Model/Controller/View



# Creating a Sharded Cluster (cont.)

```
# Now start the mongos on a standard port  
mongos --logpath "mongos-1.log" --configdb localhost:57040,localhost:57041,localhost:57042 --fork
```

- Config server keeps track of the sharding metadata (i.e. which servers are associated with which datasets)

# Creating a Sharded Cluster (cont.)

```
81 db.adminCommand( { addShard : "s0/"+"localhost:37017" } );
82 db.adminCommand( { addShard : "s1/"+"localhost:47017" } );
83 db.adminCommand( { addShard : "s2/"+"localhost:57017" } );
84 db.adminCommand({enableSharding: "game"})
85 db.adminCommand({shardCollection: "game.world", key: {X:1, Y:1}});
```

- Distribute load by queries on global coordinates, X and Y

# Resolving Concurrency Issues

How do we avoid two users modifying the same tile at the same time?

# Resolving Concurrency Issues

How do we avoid two  
users modifying the  
same tile at the same  
time?

```
17 def update_position(user,move):
18     new_pos= new_user_coord(user,move)
19
20     user_new=user.copy()
21     user_new['X']=new_pos['X']
22     user_new['Y']=new_pos['Y']
23     user_new['x']=new_pos['x']
24     user_new['y']=new_pos['y']
25     write_result=world.update(
26         {
27             "X":user_new["X"],
28             "Y":user_new["Y"],
29             "tiles":{
30                 "$not":{
31                     "$elemMatch":{
32                         "x":user_new['x'],
33                         "y":user_new['y'],
34                         "type":"rock"
35                     }
36                 },
37             },
38             "users":{
39                 "$not":{
40                     "$elemMatch":{
41                         "x":user_new['x'],
42                         "y":user_new['y'],
43                     }
44                 }
45             },
46         },{
47             "$push":{
48                 "users":user_new
49             }
50         },
51         False # no upserting
52     )
```

Filter

Update

# Thanks!

William Cross, David Percy and the rest of the Mongo Education Team  
Adam and Jae