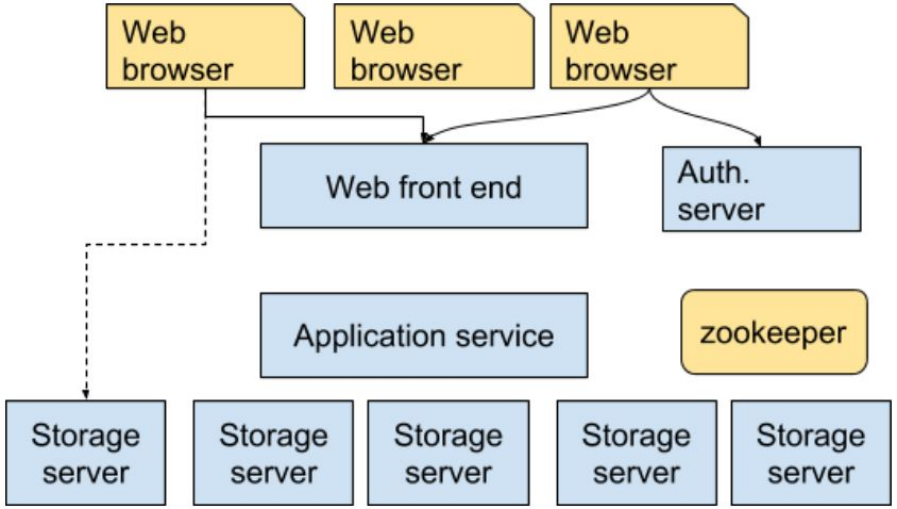
TUClery

Gkoutziouli Dimitra, Kritharakis Emmanouil, Fotakis Tzanis

Structure

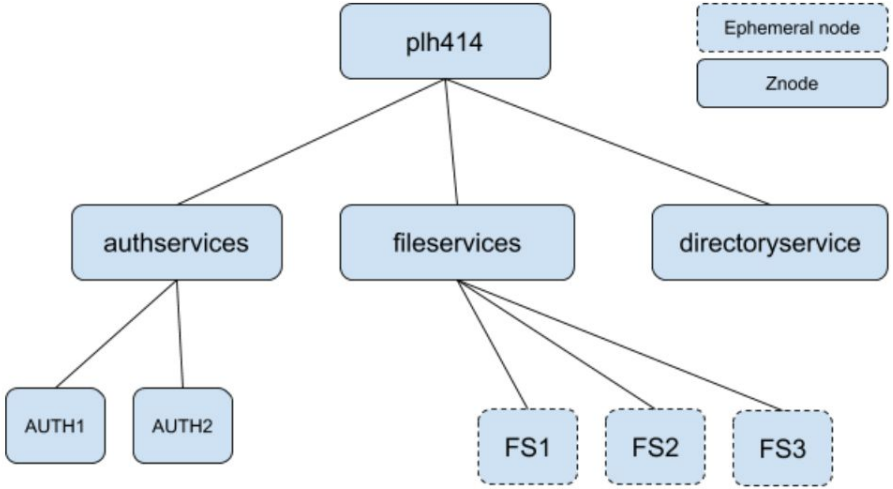
Structure

- Apache Zookeeper
- Web Service
- Application Service
- Storage Service
- Authentication Service



Structure - Apache Zookeeper

- Centralized Service
- Maintains configuration information
  - Service Name
  - Server Hostname & Port
  - Shared Keys
- Uses tree data structure
- Informs Services for changes with triggers



Structure - Web Service

- Implements the Web Interface
- Handles all HTML rendering
- Accesses the database to get/add needed data
- Implements a Rest API
  - Message Passing
  - Dynamic HTML rendering & data processing

Structure - Web Service

Implements the basic application logic:

- Login/Register
- Timeline
- Profile
- Make Friends
- Create Gallery
- Manage Gallery
- Manage Photos
- Commenting Galleries/Photos

Structure - Application Service

- Manages Storage Services
- Handles all file requests
  - Upload
  - Download
  - Delete
- Knows in which Storage Services a file exists

Structure - Storage Service

- Manages all files
  - Upload
  - Download
  - Delete
- Checks if requests are valid
  - Fingerprinting

Structure - Authentication Service

- External Login / Register
  - Using independent web-interface
- Rest API for internal Login / Register
  - Using Web Service web-interface
- Sends user data in encrypted messages

Data Bases

Each service has its own database.

- Web Service:
  - Profile
  - Friendship
  - Gallery
  - Gallery comment
  - Photo
  - Photo comment
  - Likes

- Application Service:
  - Photo

- Authentication Service:
  - User

Technologies

Technologies

Front-End

- Django 2.0
  - HTML Rendering
- Bootstrap 4
  - CSS Styling
  - Popper.js
  - jQuery.js
- FontAwesome
- Vue.js
  - Progressive JavaScript Framework
  - Dynamic HTML / Data Handling

Back-End

Back-End Handling:

- Django 2.0

Database:

- SQLite

Message Passing:

- HTTP Requests

Servers:

- Apache
- Apache Zookeeper

Additional Work since Presentation

Additional Work since Presentation

Benchmark home page

```
Server Software:        WSGIServer/0.2
Server Hostname:        127.0.0.1
Server Port:            8000

Document Path:          /
Document Length:        0 bytes

Concurrency Level:      10
Time taken for tests:   17.434 seconds
Complete requests:      1000
Failed requests:        0
Non-2xx responses:      1000
Total transferred:      228000 bytes
HTML transferred:       0 bytes
Requests per second:    57.36 [#/sec] (mean)
Time per request:       174.337 [ms] (mean)
Time per request:       17.434 [ms] (mean, across all concurrent requests)
Transfer rate:          12.77 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    0   0.0      0       0
Processing:    34  174  18.0    173     253
Waiting:       14  167  19.1    168     244
Total:         34  174  18.0    173     253

Percentage of the requests served within a certain time (ms)
  50%    173
  66%    178
  75%    182
  80%    186
  90%    192
  95%    198
  98%    207
  99%    221
 100%    253 (longest request)
```

Additional Work since Presentation

- Made use of Vue.js framework
- Dynamically downloading content (photos & comments) while scrolling down the page on home page & gallery view
- Dynamically liking & disliking photos (no page refresh needed)
- Dynamically commenting & deleting comments on photos (no page refresh needed)

Info on how to run the servers locally using the provided runServers.sh script given in the Readme.md file

Thank You!

Demo Time!