



E0-245: ASP

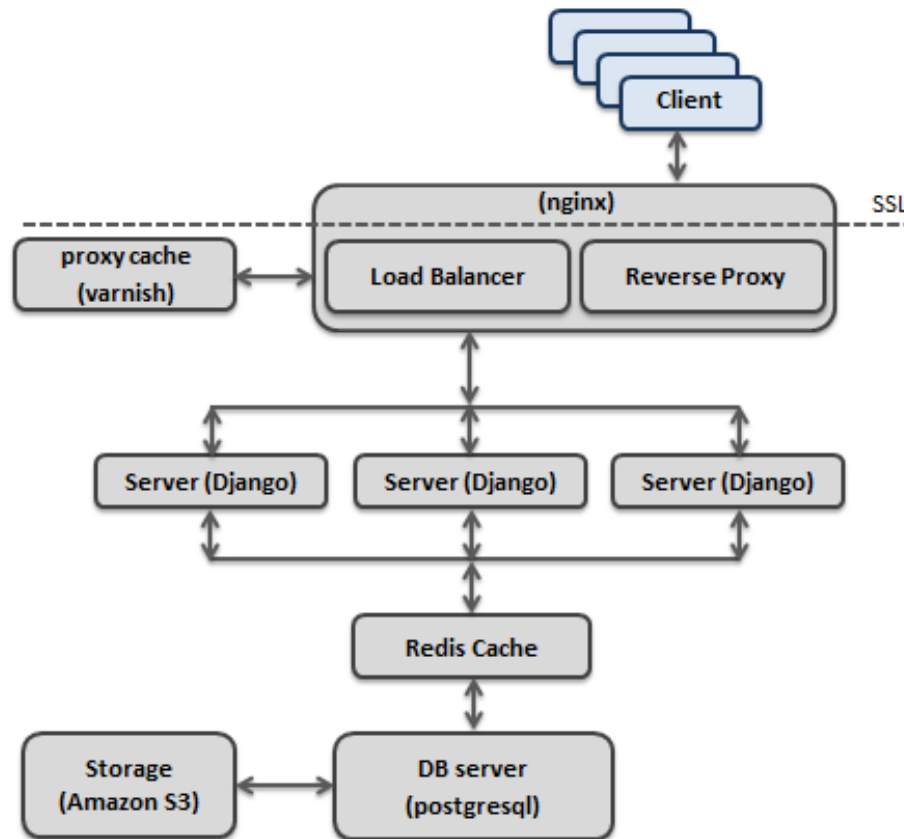
Lecture 20: Cloud Stack for Mobile Backend

Dipanjan Gope



Module 3: Hybrid Mobile-Cloud Framework

- Amazon Web Services (AWS) components: EC2, S3
- Cloud stack



Coverage

- ☒ Activity
- ☒ Views
- ☒ Intent
- ☒ ContentProvider
- ☐ BroadcastReceiver
- ☒ Service

References

- Greg Milette, Adam Stroud: Professional Android Sensor Programming, 2012, Wiley India
- Reto Meier: Professional Android 4 application development. John Wiley & Sons, 2012.

Broadcast Receiver

- Listens to `sendBroadcast(intent)`
- Must be registered in code or manifest file
- Intent Filter specifies the type of broadcasts received
- Applications DO NOT need to be active to receive intent



Broadcast Receiver

```
Intent intent = new Intent(LifeformDetectedReceiver.NEW_LIFEFORM);
intent.putExtra(LifeformDetectedReceiver.EXTRA_LIFEFORM_NAME,
                detectedLifeform);
intent.putExtra(LifeformDetectedReceiver.EXTRA_LONGITUDE,
                currentLongitude);
intent.putExtra(LifeformDetectedReceiver.EXTRA_LATITUDE,
                currentLatitude);

sendBroadcast(intent);
```

SEND

Ref: [2]

```
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;

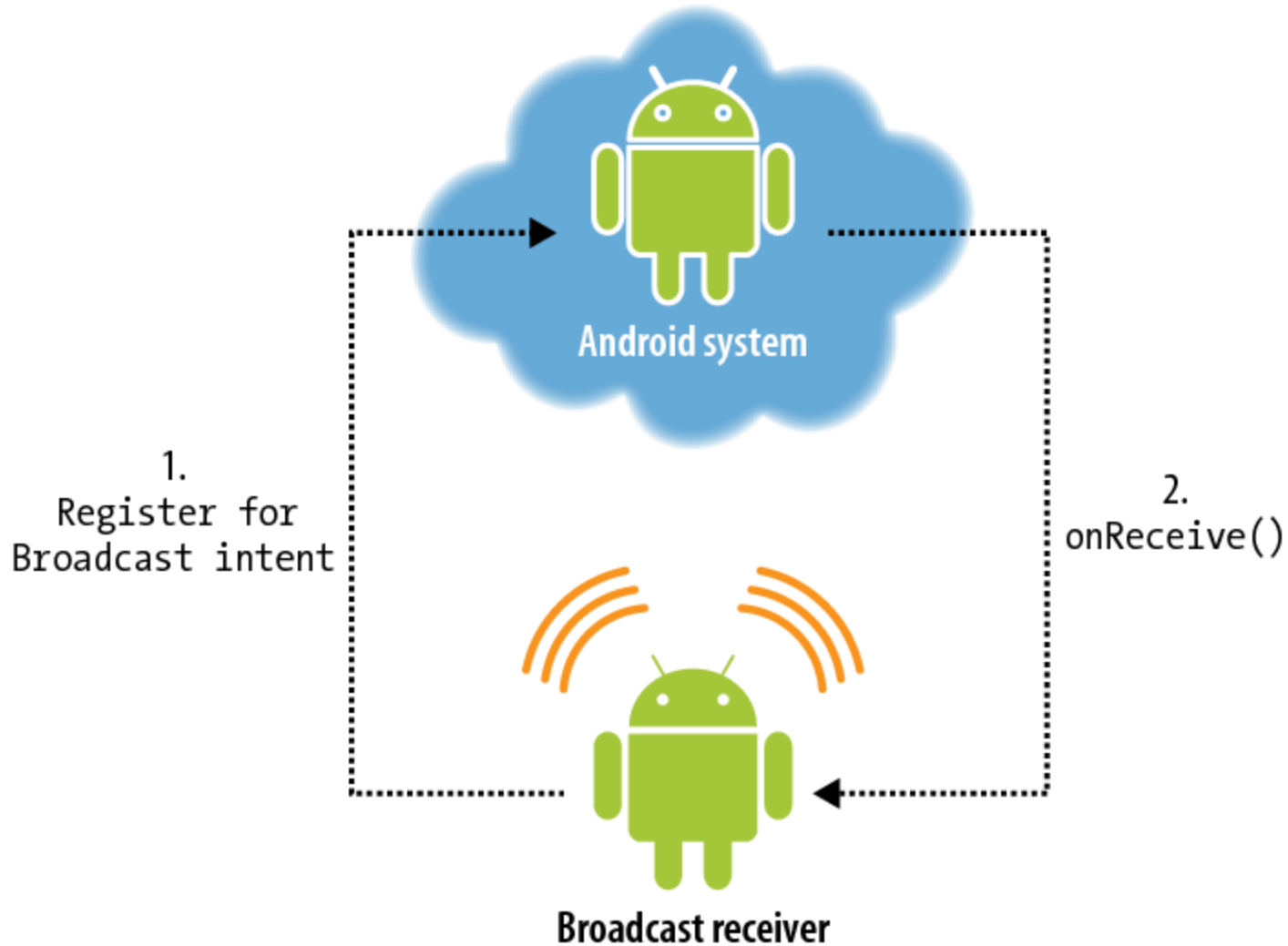
public class MyBroadcastReceiver extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
        //TODO: React to the Intent received.
    }
}
```

RECEIVE

OnReceive handler has 5 seconds to complete



Broadcast Receiver



Registering Broadcast Receiver

In Code:

```
private IntentFilter filter =
    new IntentFilter(LifeformDetectedReceiver.NEW_LIFEFORM);

private LifeformDetectedReceiver receiver =
    new LifeformDetectedReceiver();

@Override
public void onResume() {
    super.onResume();

    // Register the broadcast receiver.
    registerReceiver(receiver, filter);
}
```

```
@Override
public void onPause() {
    // Unregister the receiver
    unregisterReceiver(receiver);

    super.onPause();
}
```

Ref: [2]

Manifest File:

```
<receiver android:name=".LifeformDetectedReceiver">
    <intent-filter>
        <action android:name="com.paad.alien.action.NEW_LIFEFORM"/>
    </intent-filter>
</receiver>
```



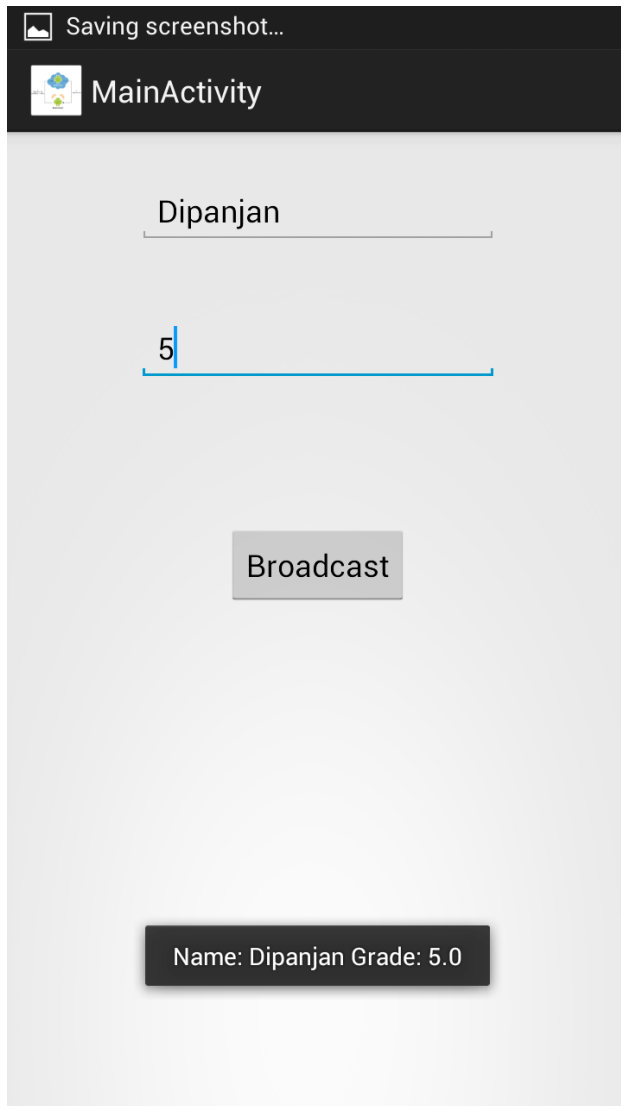
Native Broadcast Intents

- `ACTION_BOOT_COMPLETED` — Fired once when the device has completed its startup sequence. An application requires the `RECEIVE_BOOT_COMPLETED` permission to receive this broadcast.
- `ACTION_CAMERA_BUTTON` — Fired when the camera button is clicked.
- `ACTION_DATE_CHANGED` and `ACTION_TIME_CHANGED` — These actions are broadcast if the date or time on the device is manually changed (as opposed to changing through the inexorable progression of time).
- `ACTION_MEDIA_EJECT` — If the user chooses to eject the external storage media, this event is fired first. If your application is reading or writing to the external media storage, you should listen for this event to save and close any open file handles.
- `ACTION_MEDIA_MOUNTED` and `ACTION_MEDIA_UNMOUNTED` — These two events are broadcast whenever new external storage media are successfully added to or removed from the device, respectively.
- `ACTION_NEW_OUTGOING_CALL` — Broadcast when a new outgoing call is about to be placed. Listen for this broadcast to intercept outgoing calls. The number being dialed is stored in the `EXTRA_PHONE_NUMBER` extra, whereas the `resultData` in the returned Intent will be the number actually dialed. To register a Broadcast Receiver for this action, your application must declare the `PROCESS_OUTGOING_CALLS` uses-permission.
- `ACTION_SCREEN_OFF` and `ACTION_SCREEN_ON` — Broadcast when the screen turns off or on, respectively.
- `ACTION_TIMEZONE_CHANGED` — This action is broadcast whenever the phone's current time zone changes. The Intent includes a `time-zone` extra that returns the ID of the new `java.util.TimeZone`.

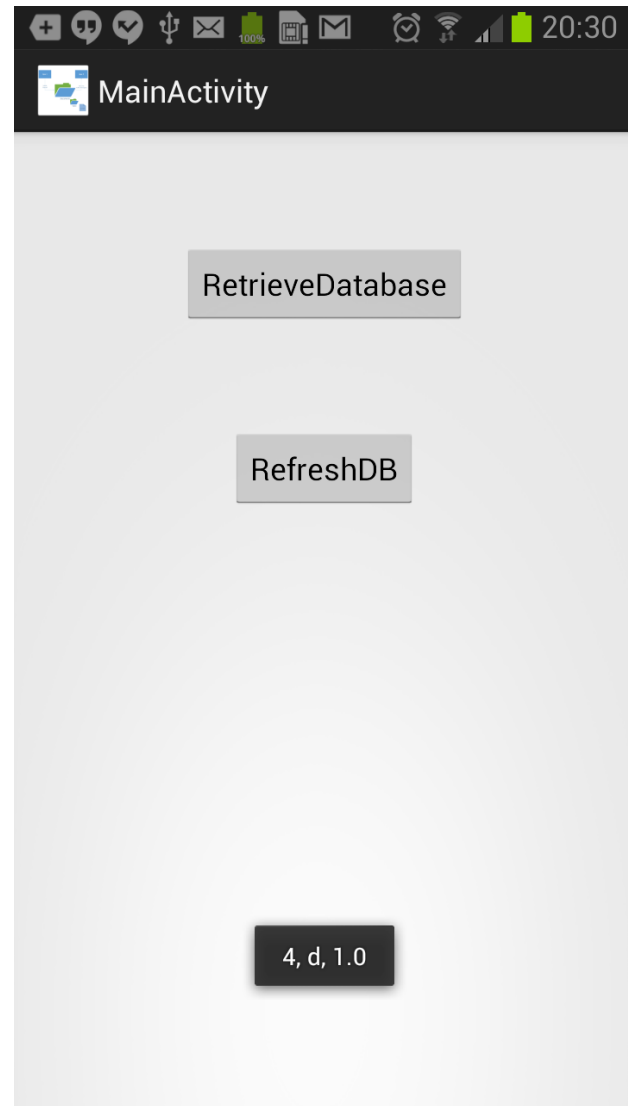
Ref: [2]



Broadcast Receiver Demo



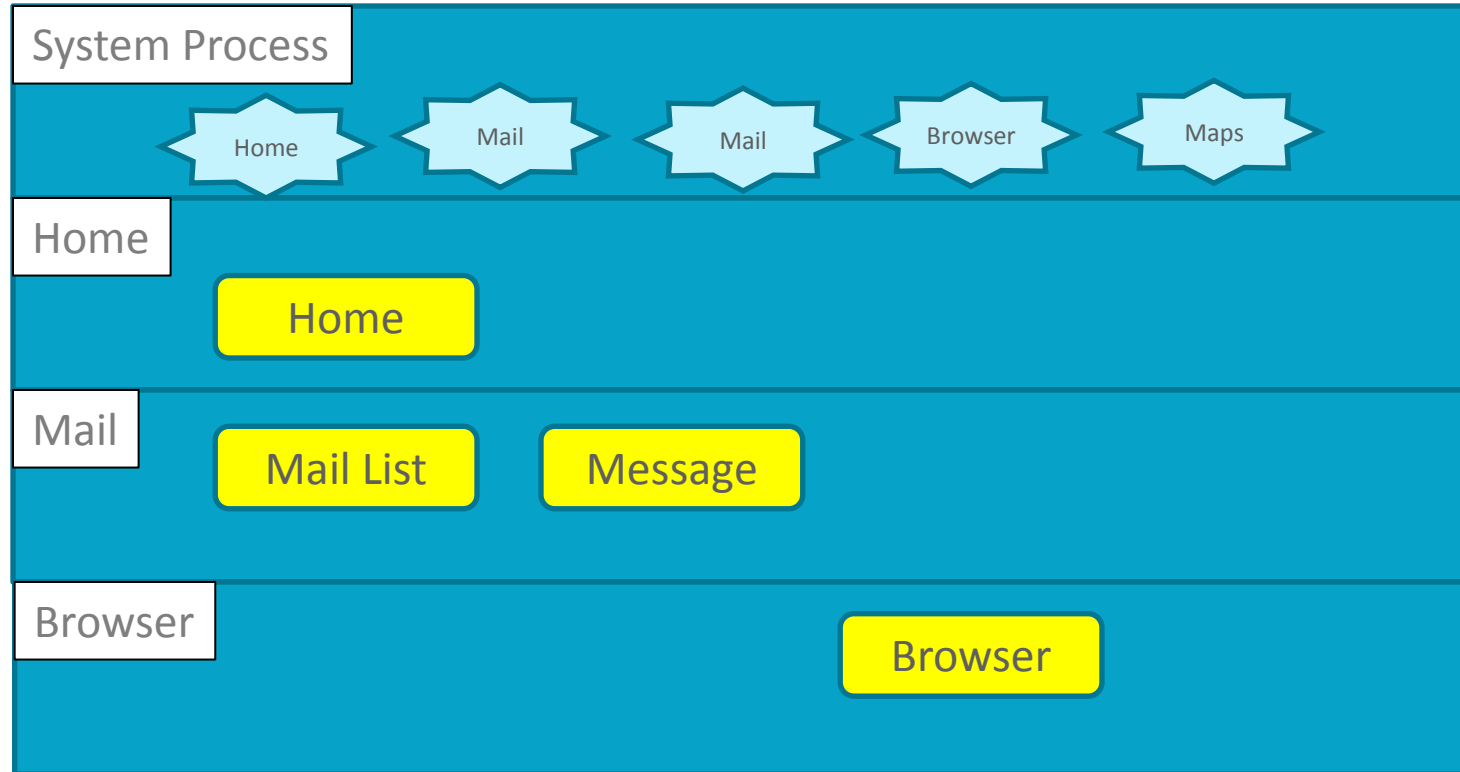
INSIDE APP



ACROSS APPS



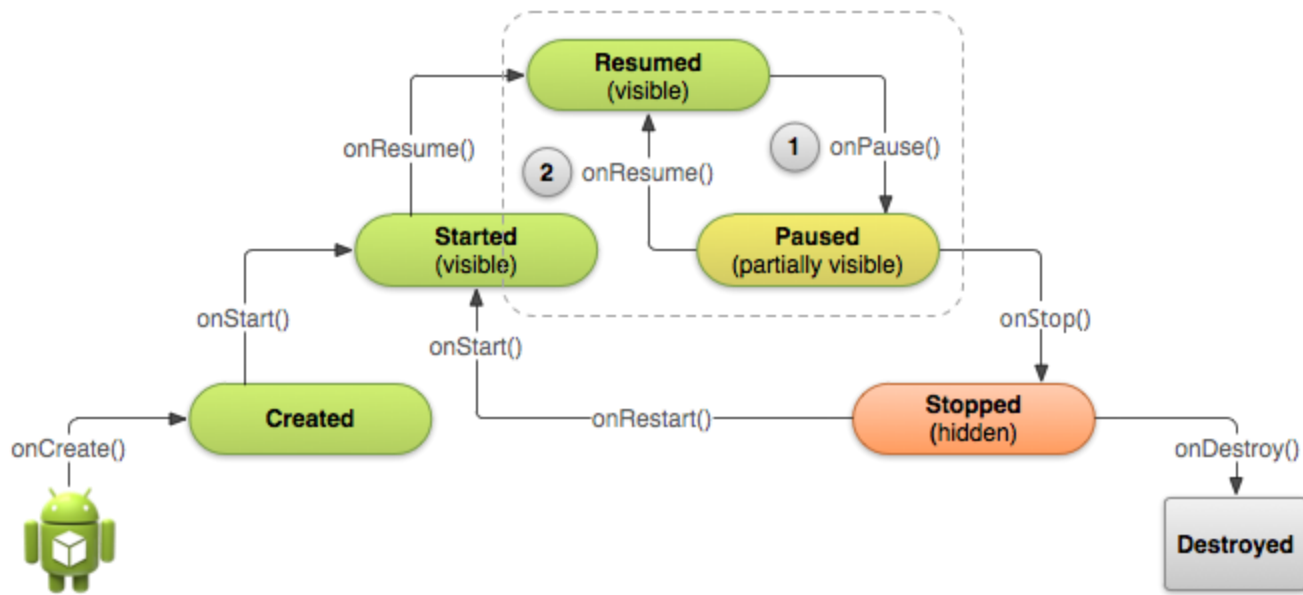
Application Lifecycle



<https://www.youtube.com/watch?v=ITfRuRkf2TM>

Androidology Part 2 of 3

Android Lifecycle



<http://developer.android.com/training/basics/activity-lifecycle/pausing.html>



Saving States

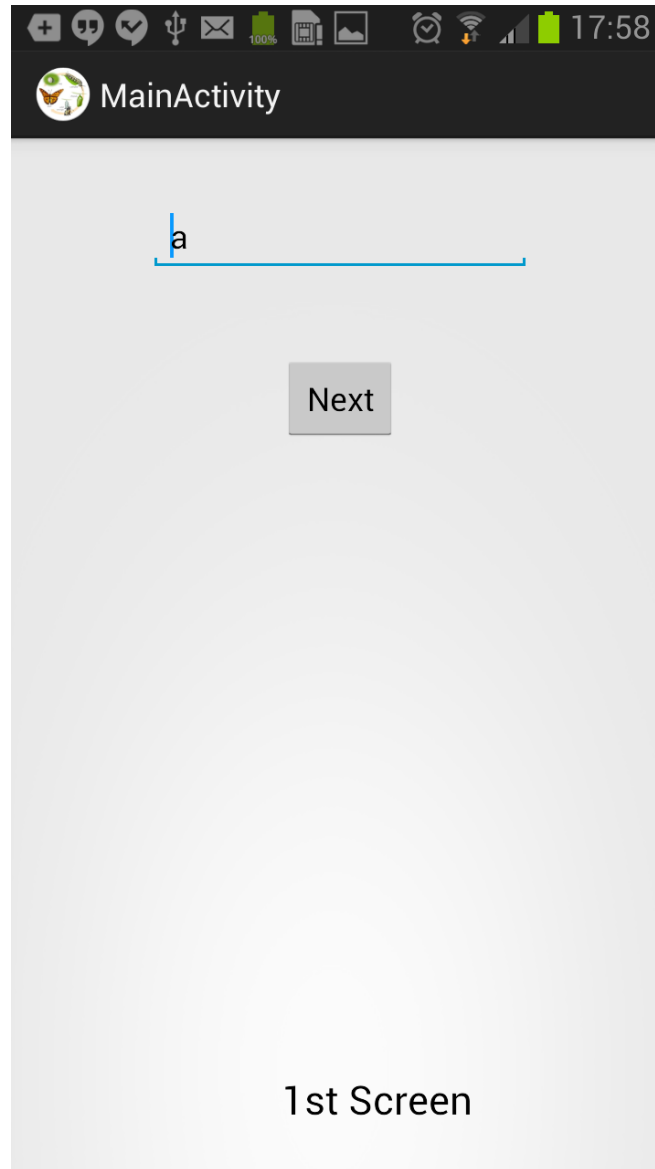
- Shared Preferences
- Saved Application UI State
- Files

```
@Override
public void onPause() {
    super.onPause();
    SharedPreferences userDetails = getSharedPreferences("userdetails", MODE_PRIVATE);
    Editor edit = userDetails.edit();
    edit.clear();
    edit.putString("username", ((TextView) findViewById(R.id.s1t)).getText().toString().trim());
    edit.commit();
}
```

```
@Override
public void onResume() {
    super.onResume();
    SharedPreferences userDetails = getSharedPreferences("userdetails", MODE_PRIVATE);
    String Uname = userDetails.getString("username", "");
    ((TextView) findViewById(R.id.s1t)).setText(Uname);
}
```



Saving State: Demo



Cloud Stack / Web Stack

- Collection of software required for web dev
 - OS
 - Programing language
 - Database
 - Web-server
 -



e.g. LAMP (Linux, Apache, MySQL, PHP)







Cloud Resources (e.g. AWS)

Amazon Web Services





Compute

-  **EC2**
Virtual Servers in the Cloud
-  **Lambda** PREVIEW
Run Code in Response to Events



Storage & Content Delivery

-  **S3**
Scalable Storage in the Cloud
-  **Storage Gateway**
Integrates On-Premises IT Environments with Cloud Storage
-  **Glacier**
Archive Storage in the Cloud
-  **CloudFront**
Global Content Delivery Network


Database

-  **RDS**
MySQL, Postgres, Oracle, SQL Server, and Amazon Aurora
-  **DynamoDB**
Predictable and Scalable NoSQL Data Store
-  **ElastiCache**
In-Memory Cache
-  **Redshift**
Managed Petabyte-Scale Data Warehouse Service





Networking

-  **VPC**
Isolated Cloud Resources
-  **Direct Connect**
Dedicated Network Connection to AWS
-  **Route 53**
Scalable DNS and Domain Name Registration




Administration & Security

-  **Directory Service**
Managed Directories in the Cloud
-  **Identity & Access Management**
Access Control and Key Management
-  **Trusted Advisor**
AWS Cloud Optimization Expert
-  **CloudTrail**
User Activity and Change Tracking
-  **Config**
Resource Configurations and Inventory
-  **CloudWatch**
Resource and Application Monitoring







Deployment & Management

-  **Elastic Beanstalk**
AWS Application Container
-  **OpsWorks**
DevOps Application Management Service
-  **CloudFormation**
Templated AWS Resource Creation
-  **CodeDeploy**
Automated Deployments




Analytics

-  **EMR**
Managed Hadoop Framework
-  **Kinesis**
Real-time Processing of Streaming Big Data
-  **Data Pipeline**
Orchestration for Data-Driven Workflows




Application Services

-  **SQS**
Message Queue Service
-  **SWF**
Workflow Service for Coordinating Application Components
-  **AppStream**
Low Latency Application Streaming
-  **Elastic Transcoder**
Easy-to-use Scalable Media Transcoding
-  **SES**
Email Sending Service
-  **CloudSearch**
Managed Search Service

Mobile Services

-  **Cognito**
User Identity and App Data Synchronization
-  **Mobile Analytics**
Understand App Usage Data at Scale
-  **SNS**
Push Notification Service

Enterprise Applications

-  **WorkSpaces**
Desktops in the Cloud
-  **WorkDocs**
Secure Enterprise Storage and Sharing Service
-  **WorkMail** PREVIEW
Secure Email and Calendaring Service

Cloud Resources (AWS)

- EC2 – Elastic Cloud Compute
- S3 – Simple Storage Service
- Elastic IP



Cloud Resources: EC2

Model	vCPU	CPU Credits / hour	Mem (GiB)	Storage
t2.micro	1	6	1	EBS-Only
t2.small	1	12	2	EBS-Only
t2.medium	2	24	4	EBS-Only

Model	vCPU	Mem (GiB)	Storage	Dedicated EBS Throughput (Mbps)
c4.large	2	3.75	EBS-Only	500
c4.xlarge	4	7.5	EBS-Only	750
c4.2xlarge	8	15	EBS-Only	1,000
c4.4xlarge	16	30	EBS-Only	2,000
c4.8xlarge	36	60	EBS-Only	4,000

Compute Optimized

Model	vCPU	Mem (GiB)	SSD Storage (GB)
m3.medium	1	3.75	1 x 4
m3.large	2	7.5	1 x 32
m3.xlarge	4	15	2 x 40
m3.2xlarge	8	30	2 x 80

Model	vCPU	Mem (GiB)	SSD Storage (GB)
r3.large	2	15.25	1 x 32
r3.xlarge	4	30.5	1 x 80
r3.2xlarge	8	61	1 x 160
r3.4xlarge	16	122	1 x 320
r3.8xlarge	32	244	2 x 320

Memory Optimized

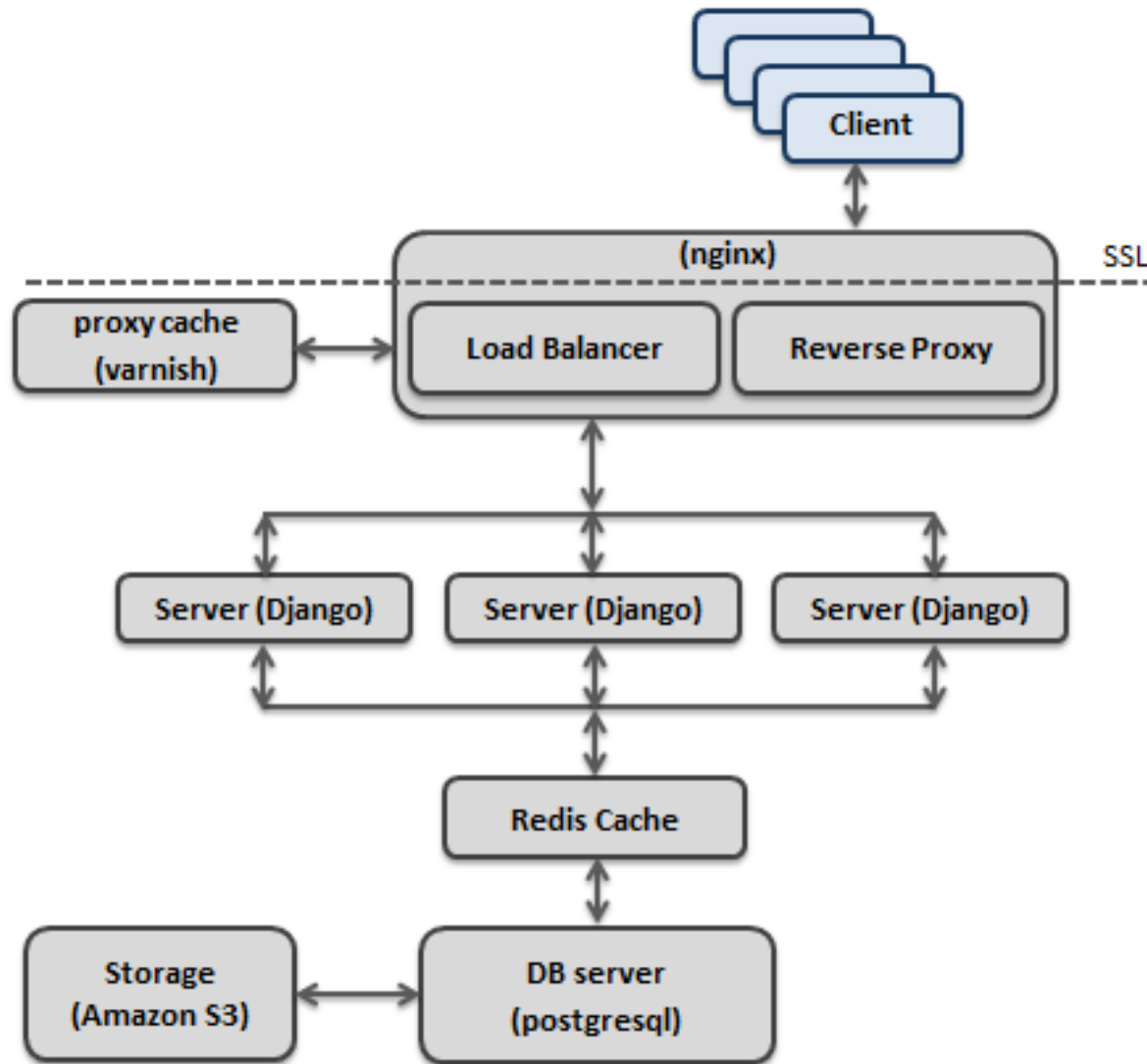


Web-stack Components

- OS/Hosting
- Load Balancing
- Reverse proxy / proxy cache
- Application/Web Servers
- Data storage
- Task Queue
- Monitoring

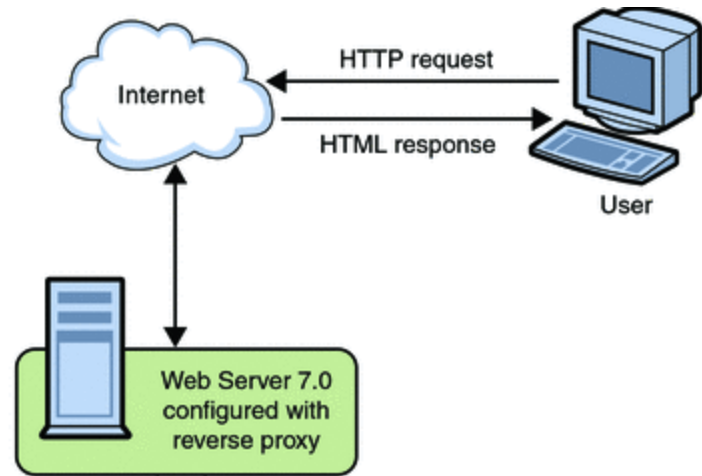


Web-stack Components



Web Server

- HTTP server
- FTP server
- Database server
- Email server



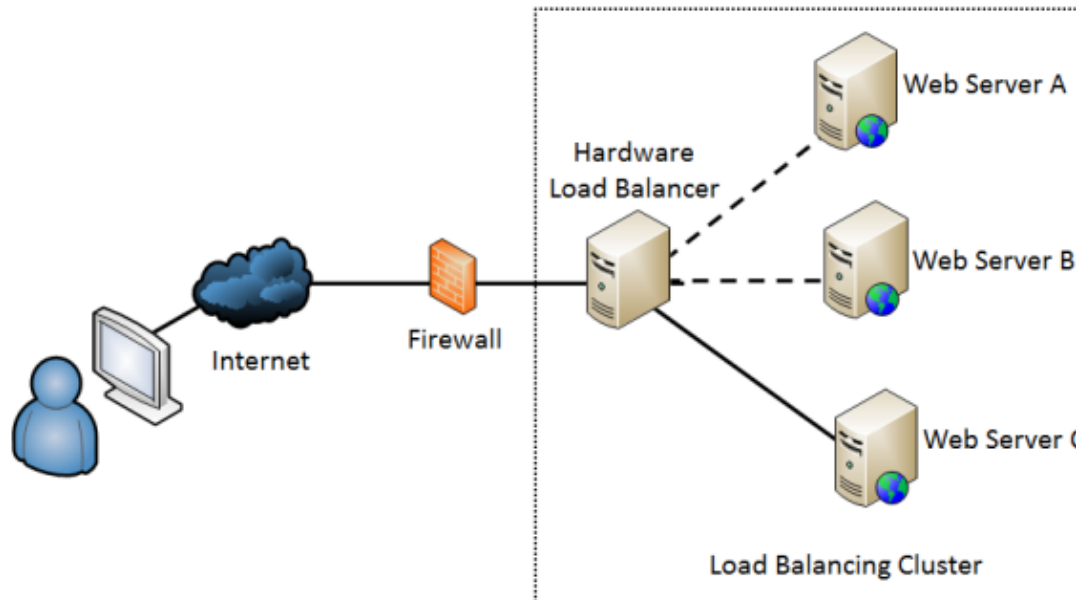
e.g. Apache, nginx, IIS



Load Balancer

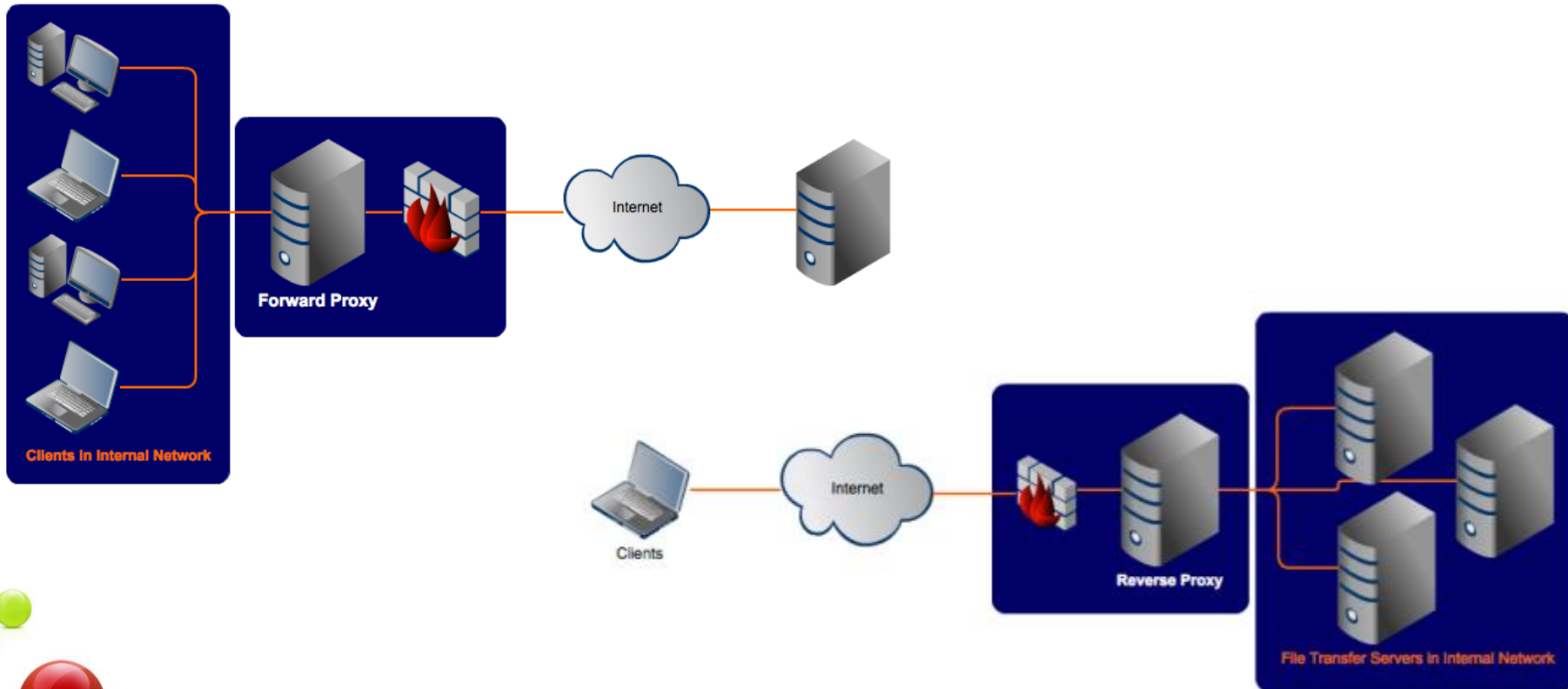
- Optimize resource use
- Maximize throughput
- Minimize response time

e.g. HAProxy, nginx, AWS ELB



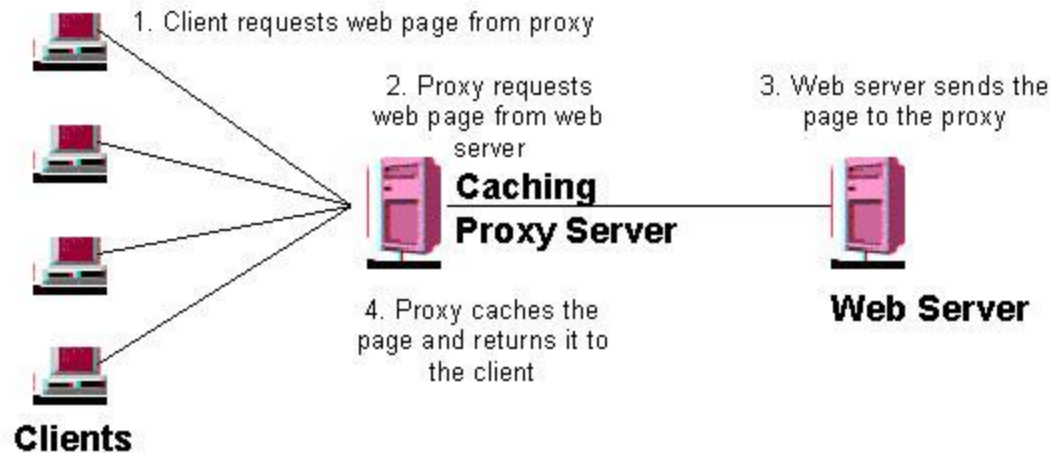
Reverse Proxy

- Proxy on behalf of server
 - Single point of access and control
- e.g. nginx, perlbal, Apache, squid, pound



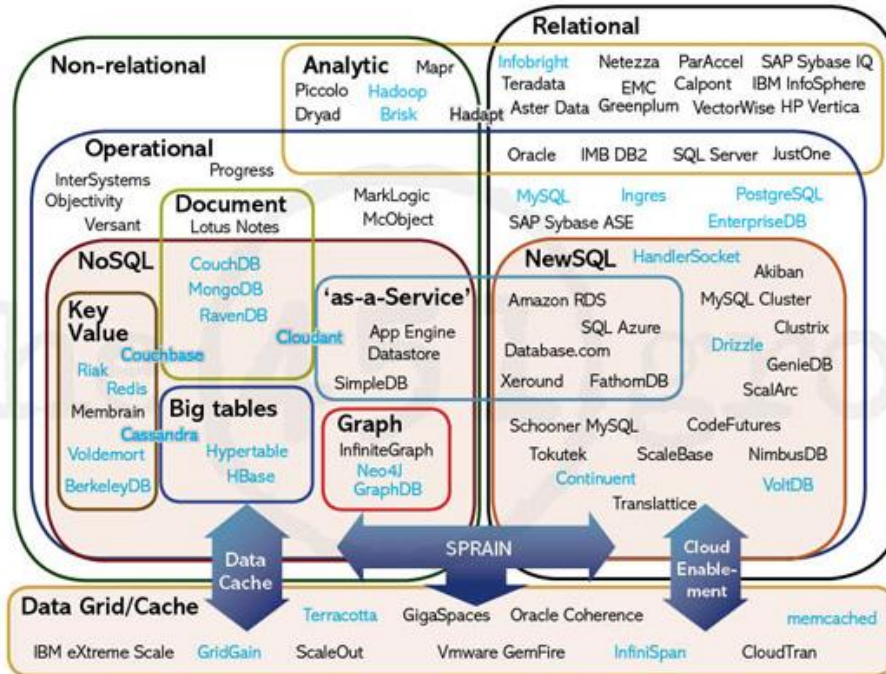
Reverse Proxy Cache

- Web Cache Types: Browser, Proxy, Reverse proxy caches
- Server acceleration
e.g. nginx, varnish



Database

- Relational DBMS: SQL (vertical scaling)
- noSQL Databases: mongoDB (horizontal scaling)



Relational vs Document data model

C1	C2	C3	C4
—	—	—	—
—	—	—	—
—	—	—	—
—	—	—	—
—	—	—	—

Relational data model

Highly-structured table organization with rigidly-defined data formats and record structure.



Document data model

Collection of complex documents with arbitrary, nested data formats and varying "record" format.

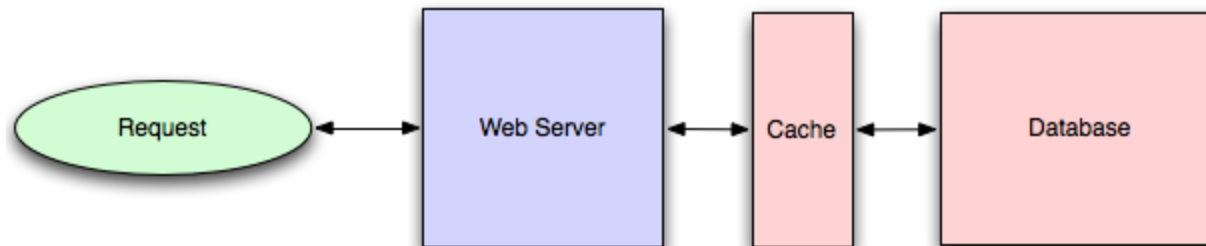
Couchbase



In-memory Cache

- RAM access is order of magnitude faster than disk

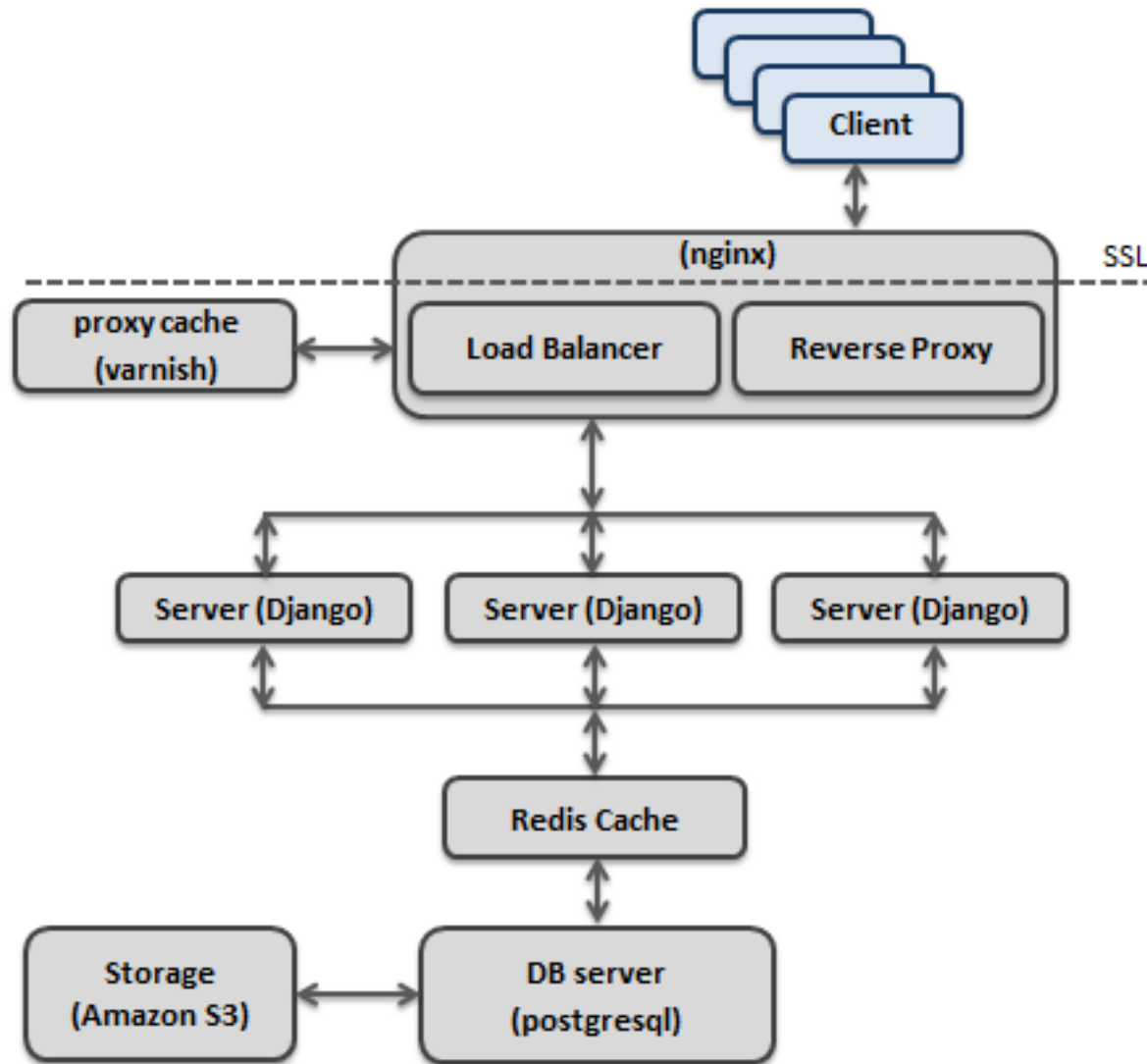
e.g. memcached, Redis



http://lethain.com/introduction-to-architecting-systems-for-scale/#application_versus_database_caching



Web-stack Components



Dates

- Final Exam: 30th April 10-1pm
 - venue to be announced on web-forum
- Final Project submission: ?
 - 15 min demo
 - Peer review
- Attendance record: ?



Grades

Time Frame	Assignment	Grading
Mar 1	Midterm	20
Apr 30	Final Exam	30
Apr ?	Final Project	40
	Class Participation	10

