Twitter with Cassandra
Team 21 - HLD Document

Adhish Singla 201403004

Saurabh Jain 201301128

Gaurav Singh 201264059

Shantanu Parashar 201202022

1 Introduction

This document contains the High Level Design descriptions of Cloud Computing Project - **" Twitter with Cassandra "**.

This document provides a complete description of all the system design and views of the Project.

The details are represented by using graphical notations such as Class Diagrams, and other supporting requirement information. All models and diagrams are built with UML Tools and represented as UML diagrams.

1.1 Scope

The software is going to be implemented as a Website. Users create verified accounts and can follow each other and also enable users to send and read short 140-character messages. Users can send personal messages to each other. Posts will be grouped based on 'Hash Tags' and Trending Topics can be seen by users.

1.2 Purpose

The purpose of this design document is to describe how the software will be structured to satisfy the requirements of the "Social Network with storage based on a Distributed Database Management System". This document will explain the design details of the system.

1.3 Intended Audience

The intended audience of this software are people who want to connect socially to the world.

2 Design in Context

In this project, object oriented approach will be used as a design method. Hence, it will be easier to implement the project and add possible future features. Since this project is a social network application, this property is critically important. Furthermore multi-layered system architecture will be used. Database, business and client layers will help modularity and adaptability of the software. With object oriented design and multi-layered architecture, portability and integrability between components will be improved.

3 Technologies

3.1 Apache Cassandra 2.1.11

Apache Cassandra is an open source distributed database management system designed to handle large amounts of data across many commodity servers, providing high availability with no single point of failure. Cassandra offers robust support for clusters spanning multiple data-centers, with asynchronous masterless replication allowing low latency operations for all clients.

3.2 Django 1.8.3

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source.

3.3 Pycassa 1.11.1

Pycassa is a Thrift-based Python client for Apache Cassandra.

Pycassa does not support CQL or Cassandra's native protocol, which are a replacement for the Thrift interface that pycassa is based on. If you are starting a new project, it is highly recommended that you use the newer DataStax python driver instead of pycassa.

3.4 Frontend

Frontend Technologies used:

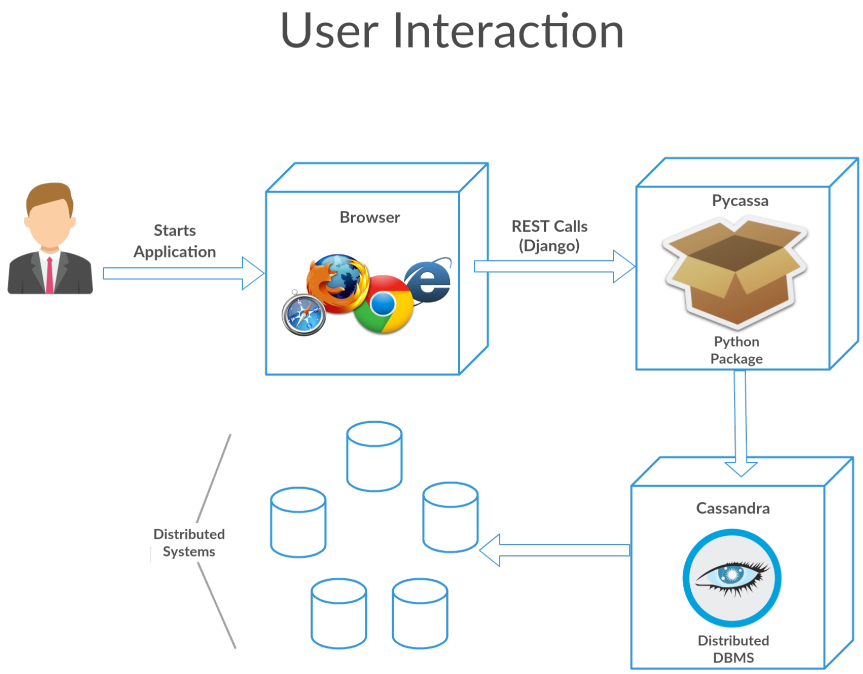
- HTML5 (Django Template)
- CSS3
- JavaScript, JQuery, AJAX

4 Project Plan

Date	Work
$14^{th}November$	Complete Functioning of Backend
$17^{th}November$	Minimal UI
$20^{th}November$	Complete UI
$21^{st}November$	Final Bug Fixes and PPT

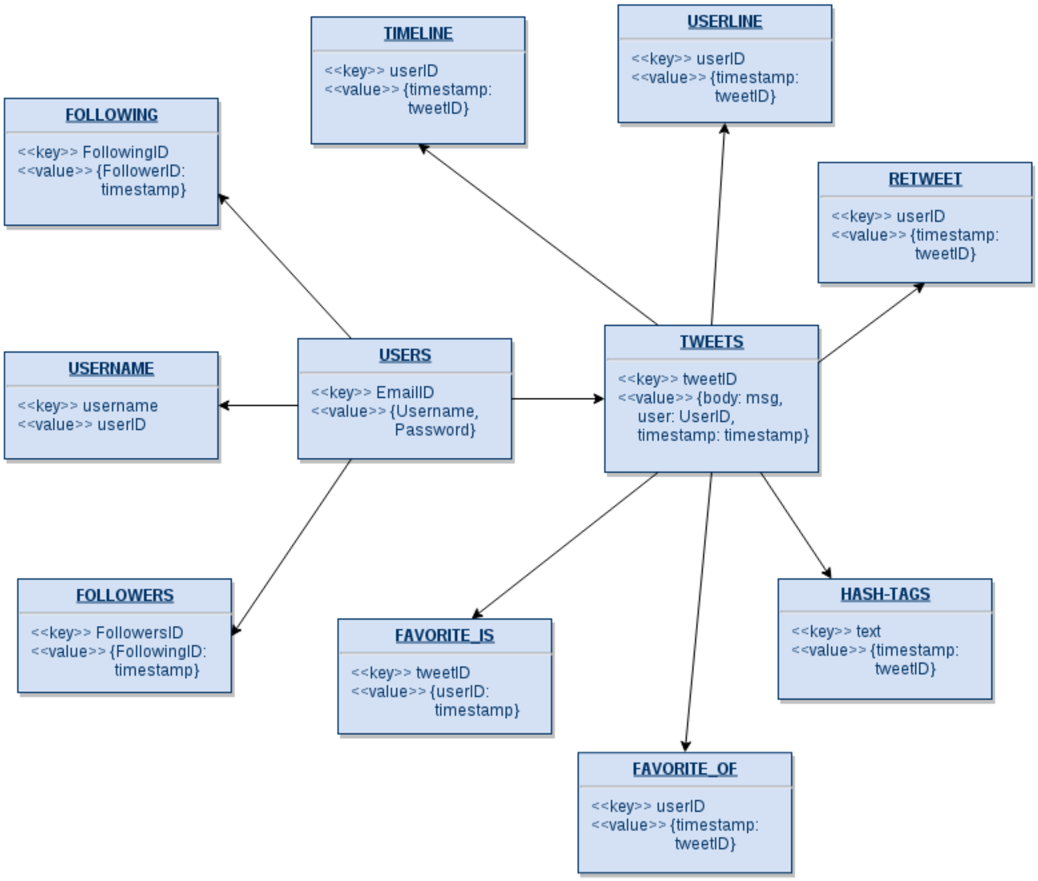
5 Workflow

1. User Interacts with the Web Application through Browser.
2. The Browser calls the REST APIs of Django based on User Interaction.
3. The REST APIs are handled by PyCassa which is a Python Package for managing Apache Cassandra.
4. Apache Cassanndra Stores and manages data through a Distributed Database.



6 Class Diagram

This is the Class Diagram of the Software describing the classes and models used.


TIMELINE
<<key>> userID
<<value>> {timestamp: tweetID}
USERLINE
<<key>> userID
<<value>> {timestamp: tweetID}
FOLLOWING
<<key>> FollowingID
<<value>> {FollowerID: timestamp}
RETWEET
<<key>> userID
<<value>> {timestamp: tweetID}
USERNAME
<<key>> username
<<value>> userID
USERS
<<key>> EmailID
<<value>> {Username, Password}
TWEETS
<<key>> tweetID
<<value>> {body: msg, user: UserID, timestamp: timestamp}
FOLLOWERS
<<key>> FollowersID
<<value>> {FollowingID: timestamp}
FAVORITE_IS
<<key>> tweetID
<<value>> {userID: timestamp}
HASH-TAGS
<<key>> text
<<value>> {timestamp: tweetID}
FAVORITE_OF
<<key>> userID
<<value>> {timestamp: tweetID}