# Engineering Data Intensive Scalable Systems

## Introduction

Internet services companies such as Google, Yahoo!, Amazon, and Facebook, have pioneered systems that have achieved unprecedented scale while still providing high level availability and a high cost-performance. These systems differ from mainstream high performance systems in fundamental ways. They are data intensive rather than compute intensive as we see with mainstream super computers spending the bulk of their time performing data I/O and manipulation rather than computation. They need to inherently support scalability, typically having high reliability and availability demands as well. Given that they often operate in the commercial space the cost-performance of these systems needs to be such that the organizations dependent on such systems can turn a profit.

Designing and building these systems require a specialized set of skills. This course will cover the set of topics needed in order to design and build data intensive scalable systems. In this domain engineers not only need to know how to architect systems that are inherently scalable, but to do so in a way that also supports high availability, reliability, and performance. Given the large distributed nature of these systems basic distributed systems concepts such as consistency and time and synchronization are also important. These systems largely operate around the clock, placing an emphasis on operational concerns. This course will introduce students to these concerns with the intent that they understand the extent to which things like deploying, monitoring, and upgrading impact the design.

The course will be a hands-on project oriented course. The basic concepts will be given during the lectures and applied in the project. The students will gain exposure to the core concepts needed to design and build such systems as well as current technologies in this space. Class size will be limited.

## Learning Objectives

Students in this class will learn what it takes to engineer systemic properties into data intensive distributed systems. Namely we will focus on:

* Scalability
* Availability
* Response Time
* Consistency
* Compute Cost

In support of these goals students will be introduced to some basic distributed systems and software architecture theory as well as relevant technologies. This is not a technology or theoretical course, however. Students will not be tested on their knowledge of theory or technology, however, rather they will be expected to demonstrate the application of this knowledge through the construction of systems with increasing expectations.

In addition to learning how to optimize on these properties in isolation, students will learn about the interplay among these concerns. Managing tradeoffs appropriately is often one of the more difficult engineering challenges and will be the focus of later projects in this course.

## Activities

There will be a series of readings, recorded lectures, and activities that the students will do on their own. Lectures will be reserved primarily for exploring more abstract complex topics in depth. We will have design exercises, discussions, and review of topics and projects during class time.

## Projects

Throughout the semester students will build a robust, scalable, e-commerce system. The system will start simply but add requirements and expectations throughout the semester. The evaluation of these projects will consist of functional tests, being able to stay within budget, and operational tests.

The kinds of concerns addressed will be things like:

* Availability: being able to develop a solution that can provide expected service despite the presence of various kinds of faults in the system (instances failing, slow instances, ect).
* Response time: being able to baseline the performance of your system, evaluate what decisions contribute to the experienced latency and improve the response time of the system.
* Scalability: ensuring the system can maintain response time goals as the number of users and volume of data increase.
* Manage multiple workloads: understand the how demands imposed by different workloads vary and design and build a system that can appropriately handle multiple workloads with varying needs.
* Consistency: understand what kinds of consistency is expected by a given context and design a system that appropriately balances the consistency expectations with other systemic concerns.
* Compute cost: be able to determine the anticipated costs associated with a given design in order to determine if the solution can be built, tested, and deployed within budget.

## Grading

Grading will be:

* Activities 30%
* Projects 60%
* Participation 10%

## Schedule

|  |  |  |  |
| --- | --- | --- | --- |
| **Week** | **Topics Due** | **Activities** | **Projects Due** |
| 10th January | * JSON Overview * JMeter Overview * AWS Setup | * Introduction |  |
| 17th January | * Client Server * N-Tiered | * Baseline Design Exercise | Project 1 (Login)  (due 22nd January) |
| 24th January | * Application State * Service Orientation | * Early Design Exercise |  |
| 31st January | * Micro Services * Availability | * State Identification | Project 2 (E Commerce App)  Due 5th Feb |
| 7TH February | * Performance | * Availability Quiz |  |
| 14TH February | * Scalability | * Design for Availability | Project 3 (Availability)  Due 26th Feb |
| 21st February | * Post-SQL * Designing Data Model | * Scalability Quiz |  |
| 28H February | * Consistency * Global State & Time | * Post-SQL Quiz | Project 4 (Scale Data)  Due 22nd March |
| 14th March | * Distributed Transactions * Concurrency | * Data Modeling Exercise |  |
| 21th March | * Intro to the Cloud * Availability in the Cloud | * Scalability Design Exercise |  |
| 28th March | * Security | * Project Discussion | Project 5 (More Scalability)  Due 16th April |
| 3rd April | * Performance in the Cloud | * Security Quiz |  |
| 10th April | * Serverless Architecture | * ThanksGiving/No Class |  |
| 17th April | * Operational Concerns | * Project Discussion |  |
| 24th April | * Case Study | * Wrap up |  |