GRADEBADGE

DEVELOPMENT OF A CLOUD-BASED REWARD APPLICATION

A Project

Presented to the

Faculty of

California State University,

San Bernardino

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

in

Computer Science

---

by

Erwin Toni Soekianto

April 2013

GRADEBADGE

DEVELOPMENT OF A CLOUD-BASED REWARD APPLICATION

A Project

Presented to the

Faculty of

California State University,

San Bernardino

---

by

Erwin Toni Soekianto

April 2013

Approved by:

David Turner, Chair, Computer Science and Engineering	Date
Richard J. Botting	
Arturo I. Concepcion	

ABSTRACT

The purpose of this project is to investigate the use of cloud-based service to deliver cutting-edge application. For this purpose, the prototype of reward application using badges will be developed to illustrate the emerging paradigms, including the Node.js, NoSQL database using MongoDB, cloud computing using Heroku, and mobile devices.

In additional to Node.js, web technologies might be used such as HTML (5), Javascript, and CSS, and jQuery Mobile to serve as clients to this application. We would also be using cloud-based source control and repositories, GitHub, which would allow automation in deploying the application.

The goal of this application is to help any organizations or groups to interact with their members in the fun way. It would keep the members engaged by giving badges as rewards for effort or achievement they have done. In order to gets this application going viral quickly; it would be integrated to social networking sites like Facebook.

ACKNOWLEDGEMENTS

I would like to thank all the people with whom I have worked while pursuing my master's degree at California State University, San Bernardino (CSUSB). I wish I could list all their names but the list would be too long and I would still probably leave some people out. Studying in the School of Computer Science and Engineering at CSUSB has been a tremendous learning experience, both personally and professionally.

Thank you to the following faculty of the Computer Science and Engineering department for their invaluable guidance, advice, support, help, and patience during this project's long gestation: Dr. David Turner, Dr. Arturo Concepcion and Dr. Richard Botting.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

1.1 Background

A long time ago, businesses used to produce their own electric power. And due to engineering breakthrough in electric generator and transmission method, it became easier to produce and transmit electricity, to supply businesses that once produced their own electricity. As more businesses started buying electric power, making utility expanded and electricity cheaper.

And today, just like the utilities, instead of buying servers to run your websites or applications, you rent servers or server spaces from cloud computing providers. Just like renting an apartment, even you are in the same building with other people, you still have your own space. As more people rent and buy computing power, making clould computing expanded and very popular.

Today there are many cloud computing providers and they are providing different type of services. Some may provide hosting, database, code repositories or storage or combinations. Few famous providers are Google App Engine, Windows Azure, Amazon Web Services and Heroku.

Google App Engine provides infrastructure to build the web application on the same scalable systems that power Google applications which support Phyton, Java, PHP and Go programming language. Google App Engine also provides several options for storing data, using App Engine Datastore, Google Cloud SQL, and Google Could Storage. Windows Azure provides similar service as Google App Engine but supports different set of programming language, such as .Net, Java, Node.js and Phyton.

Amazon provides scalable cloud computing, which allow users to choose what type of operating system and configuration of the servers they need, but it can scale as needed. It is more flexible to use any technologies, but require a lot of time and expertise to set it up.

In this project, Heroku is used to as cloup application platform which support Node.js, Ruby, Clojure, Java, Phyton and Scala. Heroku lets you use and publish an application that people can use right away with no cost and obligation, and you can take advantage of the same scalable technologies that Facebook applications are built on, and the reliability, performance and security.

Among all the programming language supported in Heroku, in this project, Node.js is used as main programming language in the server side. HTML5, Javascript, Jquery and Bootstap framework will be used in the client side. As data store provider, MongoLab is used which support MongoDB database.

Node.js and MongoDB in Heroku are often used together for scalable web technology. The following describe the clould computing service providers used in this project.

1.2 Facebook

Facebook is a very popular socila networking website and has billions of users. It has proven to be good platrom to use for web application and take advantage of its social networking, to connect to other Facebook users. Facebook also allows other application to access the user's data with their authorization using its API. [7]

1.3 Heroku

Cloud computing is a model which makes use of computer hardware and software that are accessed through the Internet as services. There are several choices of cloud

computing services available, but for this project we choose the one provided by Heroku, the cloud computing partner of Facebook. [11].

The reasons are that Heroku lets you use and publish an application that people can use right away with no cost and obligation, and you can take advantage of the same scalable technologies that Facebook applications are built on, and attain a similar level of reliability, performance and security.

1.4 MongoDB

There are many different types of cloud-based datastore services to choose from. For this project we will use MongoDB, as it works well Node.js and Heroku. MongoDB is a scalable, high-performance, open source, NoSQL document-based database. MongoDB features include document-oriented storage, indexes, replication, high availability, auto-sharding, and querying. [20]

1.5 MongoLab

MongoLab is the cloud computing provider for MongoDB database, easily integrated with Heroku. [21]

1.6 Git

Git is a distributed version control system. This project uses git with GitHub, a cloud-based provider of remote git repository storage. Heroku uses git as a means to deploy web applications to its servers. Git allows easy creation of testing, staging, and production versions of the application. [8]

1.7 Bootstrap

Bootstrap by Twitter provides responsive design framework that work well for application to be used in desktop, tablet and mobile phone. The UI of this project use Bootstrap. [4]

1.8 Jquery

Jquery is used for AJAX and DOM manipulation. [18]

1.9 Node.js

Cloud-based services support apps written in several different programming languages, such as Java, Python, PHP, Javascript, Ruby and many more. For this project we would use Javascript running in a Node.js context. Node.js is a platform built on Chrome's JavaScript runtime for easily building fast, scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices. [22]

1.10 Purpose

To explore the new technologies, to create cross-platform reward application that individual can use.

1.11 Project Scope

Project does not include database sharding features to allow greater degree of scalability.

The GradeBadge application provides the following functionalities:

- Create group
- Create Badge
- Add Member
- Issue Badges to Members
- View Badge Earned
- Share Badge to Social Networking

1.12 Related Work

There is another similar project in cloud computing, it used Google App Engine instead of Heroku as cloud computing providers, the main programming language used is Java with Google data store instead of Node.js and MongoDB. It uses Jquery Mobile as UI framework instead of BootStrap. [31] So there has been some increasing interest in cloud computing technology, and there are dozens other famous application that use similar technologies.

1.13 Project Limitations

Users must have Facebook account, logged in to facebook and authorized access to basic information (name, profile picture and friend list) .For best experience must use modern browser in either PC or tablet or smart phone.

1.14 Definitions, Acronyms, and Abbreviations

The definitions, acronyms, and abbreviations used in the document are described in this section.

- GradeBadge: The name of this project

- API: Application Programming Interface is a set of routines that an application uses to request and carry out low-level services performed by a computer's operating system; also, a set of calling conventions in programming that defines how a service is invoked through the application [9].
- Cloud computing: Cloud computing is the use of computing resources (hardware and software) that are delivered as a service over a network (typically the Internet) [**?**].
- JQuery: A javascript library provided by JQuery for building web based applications [18].
- UI: User Interface
- CSUSB: California State University, San Bernardino.
- HTML: HyperText Markup Language is the authoring language used to create documents on the World Wide Web [28].
- HTTPS: Hyper Text Transfer Protocol Secure is a secure network protocol used to encrypt data transferred between server and client [13].
- MVC: Model-View-Controller is an architectural pattern used in software engineering to isolate business logic from user interface considerations [32].
- UML: The Unified Modeling Language is the industry-standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems [27].
- Microsoft Azure: Cloud Computing platform provided by Microsoft [30].
- Google App Engine: Cloud Computing platform provided by Google
- Amazon Web Services: Cloud Computing platform provided by Amazon [2].

- Heroku:Cloud Application platform provided by Heroku [11].
- Android : Mobile Operating System provided by Google [3].
- IOS : Mobile Operating System provided by Apple [15].
- NoSQL: Uses key-value pairs for storing data unlike traditional Relational Database Management [23].
- JSON : Javascript Object Notation built using key and value pairs [19].
- Ajax: Asynchronous JavaScript and XML/JSON format for communicating from client to the server [1].
- OOP : Object Oriented Programming concept with objects representing real world entities. Methods expose state of the object [29].

2. SPECIFIC REQUIREMENT

2.1 External Interfaces Requirement

2.1.1 Hardware Interfaces

The application will be hosted in Heroku server. The web server is listening on port 80. The system is a web based application; clients are requiring using a high speed Internet connection and using a up-to-date web browser.

2.1.2 Software Interfaces

JavaScript will be implemented throughout the website in order to display the correct feature the user requested. And HTML5 may be implemented throughout the website in order to display the correct feature the user requested.

2.1.3 Communication Interfaces

This application is designed to be viewed on any internet browser provided that, if JavaScript and images features are enabled. And the browser is HTML5 compatible. Performance may vary slightly between browsers. However, the functionality of the site should not be impaired.

2.2 Functional Requirement

The function specified on this on this section directly correspond to work that will be conducted on this project, as shown in the use case diagram in Figure 2.1.

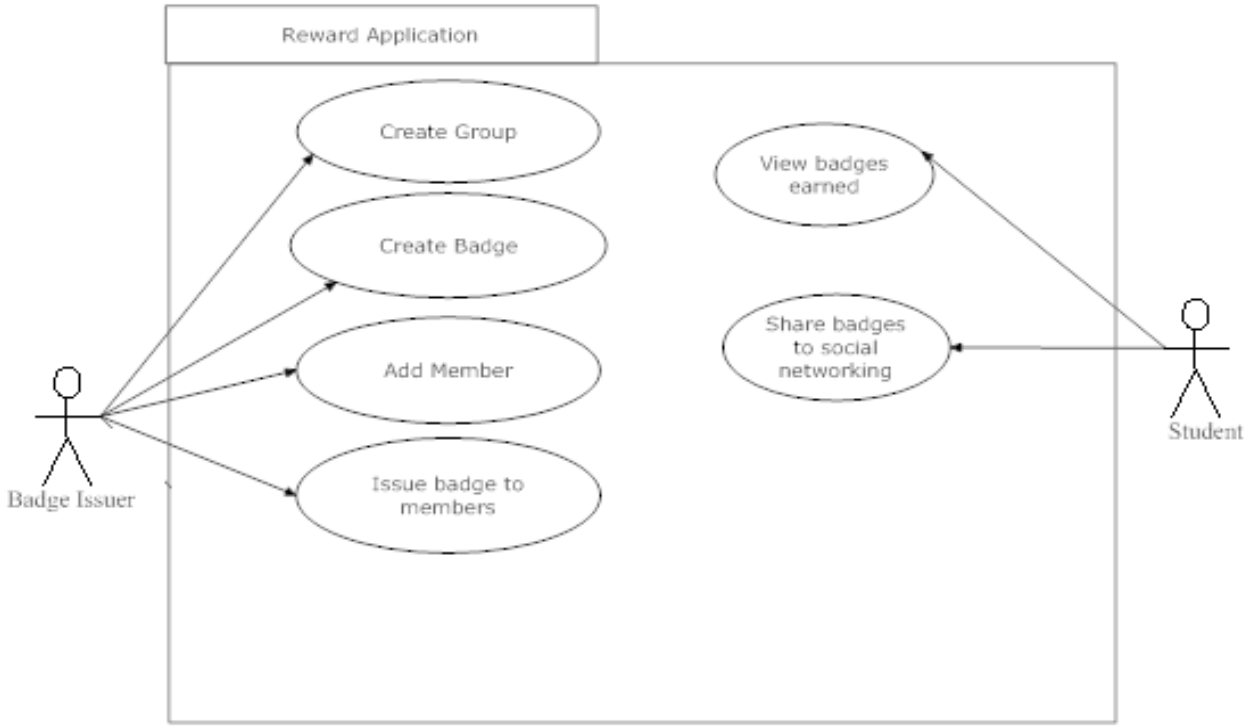

Fig. 2.1: Use Case Diagram

2.2.1 Create group

This functionality allows organizers or badge issuers to create groups, which will have a set badge collection.

2.2.2 Create Badge

This functionality allows badge issuers to create badge to be added to badge collection. A badge would have at least badge name and description.

2.2.3 Add Member

This functionality would allow organizers or badge issuers to add members into the groups as badge recipients. Member would have at least email address and name.

2.2.4 Issue Badges to Members

This functionality would allow badge issuers to issue badges to members.

2.2.5 View Badge Earned

This functionality would allow badge recipients to view all the badges that they have earned.

2.2.6 Share Badge to Social Networking

This functionality would allow badge recipients to share the badge to their social networking site such as Facebook

2.3 Performance Requirement

This application is going to be hosted in the Heroku cloud server, so the performance of this application would be high.

2.4 Design Constraint

This application requires internet-enabled devices and internet connection to perform. And every user must have Facebook account to be able to use this.

2.5 Software System Attributes

The author will keep coding standard with proper commenting and documentation.

3. SYSTEM ARCHITECTURE

3.1 Overview

This application uses HTTPS exclusively for security reason, except in local developer environment we use HTTP, as shown in Figure 3.1.

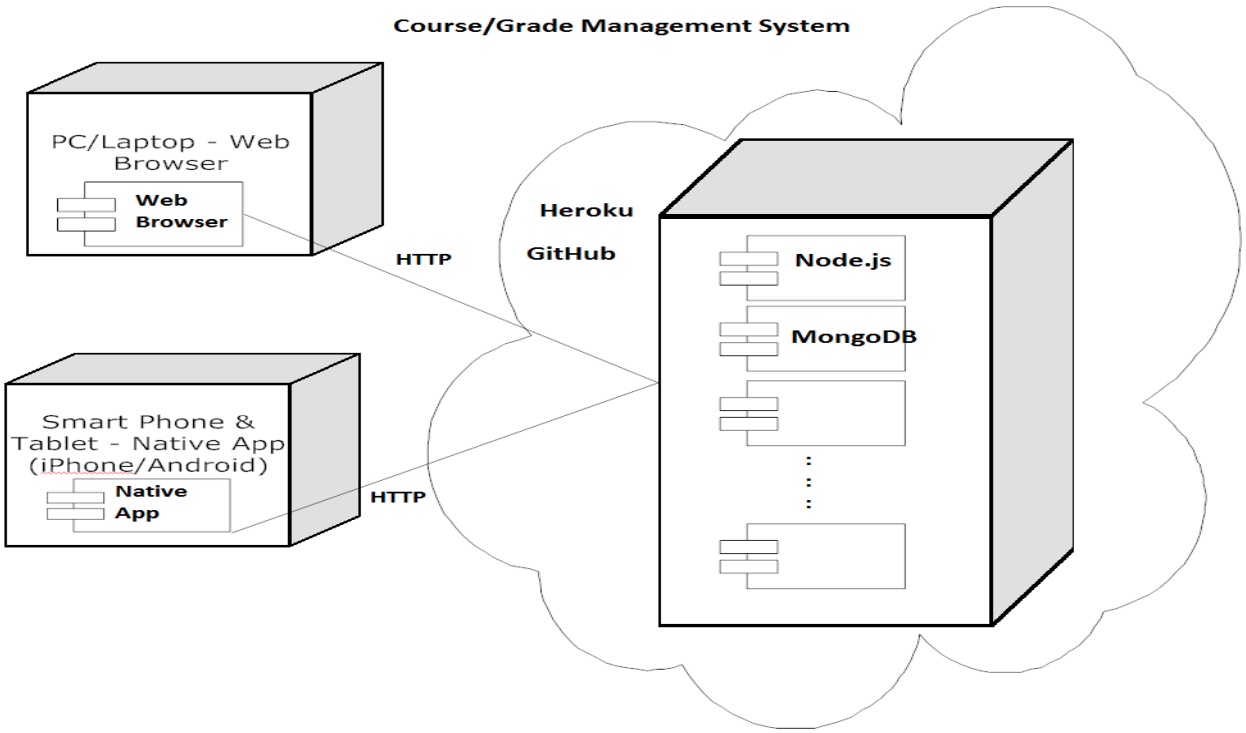

Fig. 3.1: Deployment Diagram

3.2 Deployment Workflow

There are three type of environments used in the deployment workflow: Development, Staging and Production. And this is how the workflow will look like in Figure 3.2.

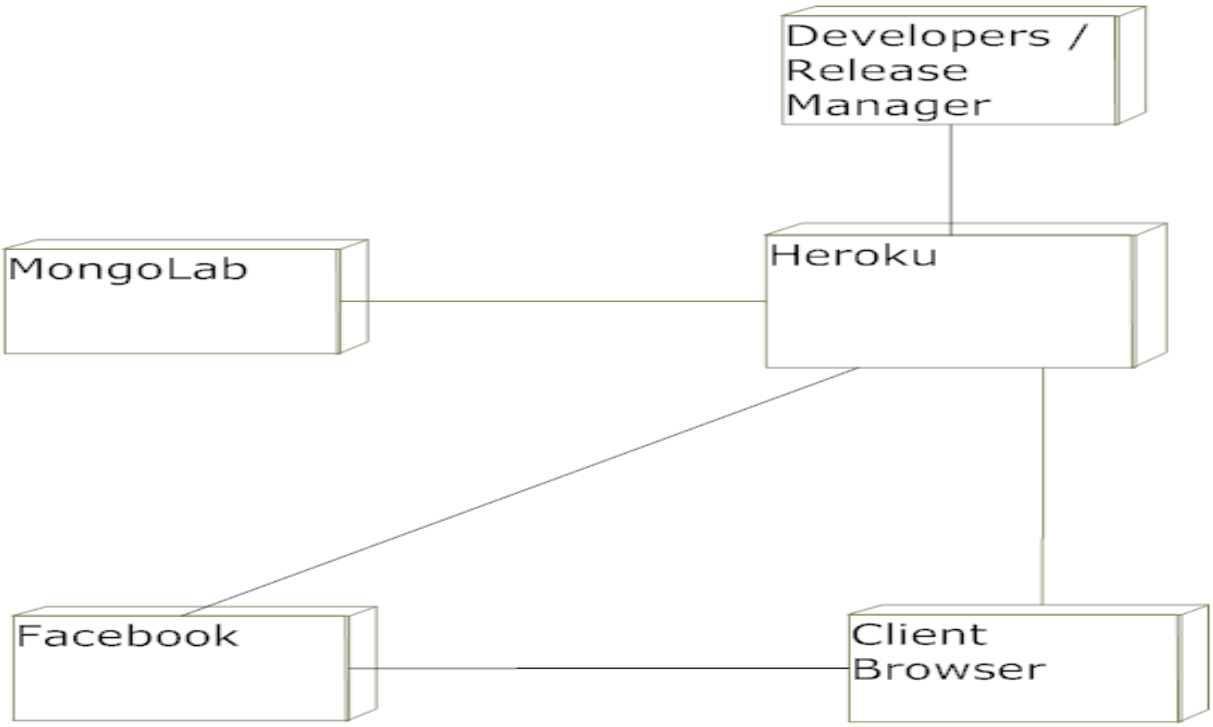

Fig. 3.2: System Integration

- Developers work on new features or bugs fixing in development branch. Only minor updates are committed directly to stable development branch.
- Once features are implemented and/or set of bugs are fixed, they are merged in to staging branch and deployed to staging environment for testing and quality assurance
- After testing is completed, the snapshop of staging branch is kept for prduction

deployment, otherwise the process will repeat until the testing is completed.

- On the release date, the working staging branch is deployed to production environment.

On this project, git is used as code repositories, to manage developments, staging and production branch. And Heroku toolbelt is also used to set the enviroment config variable for each deployment. Heroku allows users to use git to deploy automatically from local repositories.

3.3 Heroku

In this project there are two sets of heroku instance used, staging and production. Heroku connects to MongoLab using Mongo Protocol to get and/or write the data to database, and Heroku also talks to Facebook server via Open Graph API.

3.4 MongoLab

In this project there are three sets of mongo database used, development, staging and production. It is important to keep the versions of database since new version of changes may include changes in database structure, so rolling back or forward the application version would not cause any error.

3.5 Facebook

Facebook is playing an important role in this project. Facebook provides user authentication and social media integration. Facebook allows connection using Facebook API and Open Graph API.

3.6 Client Browser

Client browser uses HTTPS GET for static content, and HTTPS POST for AJAX request to Heroku. And client browser also connects to Facebook server directly using Facebook API and Open Graph API in HTTPS.

4. SYSTEM DESIGN

4.1 Design Overview

All incoming AJAX requests are submitted using HTTP POST and contain data encoded using JSON.....

4.2 Model View Controller Architecture

This application is based on Model View Controller (MVC) Architecture. Model View Controller (MVC) architecture is a software design pattern for separating different components of a software application. [32] There are three main categories in the MVC architecture:

- Model : it represents data in the application. All the business rules are handled in the model.
- View: it represents UI components in the application. The UI components are responsible for presenting the model and for collecting user inputs.
- Controller: it is responsible for updating the data in the model and notifying the view about changes in the model.

There are two sections of this project, Server-side and Client-side as described in the following

4.3 Server-side Architecture Design

All node js modules that start with req_*.js are request handler that get requested from router.js. All ajax requests will go through req_op.js, that verifies the user logged-in to Facebook and app_version is current. Every req_op.js request must contains Facebook access_token and app_version. If the user is not logged-in to Facebook, req_op.js returns the following JSON document, login:true, if the version is not current, then req_op.js return the following JSON document, ver:true.

- .env : This is the setup file that contains environment variables. This file only exist in developer local environment, and these values in this file would be set in each Heroku environment config for staging and production.
- .gitignore: This is the setup file that contains list of files or folders that will be ignored when committing or pushing to git repositories.
- .slugignore: This is the setup file that contains list of files or folders that will be ignored when calculating the slug limit in Heroku
- package.json: This is the setup file that contains of list of dependencies and engine version use in the application. This file also contains application name, version and description.
- Procfile: This is the setup file that tells Heroku how to launch the application
- main.js: This is the main module in node js, which contains the code that verify all neccessary environment variables are set correctly. It also invoke initialization in neccesarry modules to start the application, after the initialization is completed, it starts the HTTP request handling loop.
- router.js: This module routes incoming requests to the right module.
- app_ajax.js: This module contains the application wide AJAX handling routines

- app_http.js: This module contains all of HTTP protocol routines for the application, caching headers, compression header and other HTTP based optimization are implemented in this module.
- fb.js: This module contains all code that interact with Facebook.
- logger.js: This module contains application wide logging functionalities.
- model.js: This module initializes the database connnection pool during server start
- req_app.js: This module handles request for application HTML template for badge earner
- req_counter.js: This module handles request for logging counter
- req_file.js: This module handles request for static content.
- req_issuer.js: This module handles request for application HTML template for badge issuers
- req_mem.js: This module handles request for memory usage
- req_root.js: This module handles request for static content under the root URL
- req_op.js : This module handles all AJAX request from client and routes to appropriate modules.

4.4 Mapping of Model Classes to MongoDB

There will be one node js module to represent the mongoDB collection, named model_(collection_name).js. And many-to-many relationships are represented by linking documents, named (a)_(b)_links

- model_group.js: this node.js module represents Groups collection

- model_badge.js: this node.js module represents Badges colletion
- model_user.js: this node.js module represents Users collection
- model_group_admin.js: this node.js module represents group_admin_links collection
- model_group_member.js: this node.js module represents group_member_links collection
- model_user_badge.js: this node.js module represents user_badge_links collection
- model_group_badge.js: this node.js module represents group_badge_links collection

4.5 Request Handler Operation

There will be one node js module to handle ajax request from client, named op_(request).js. Every request may read, write or update to and from more than one collection

- op_read_badges_by_group.js
- op_read_groups_by_admin.js
- op_save_badge.js
- op_save_group.js
- still more.....

4.6 Client-side Architecutre Design

- app.html : This is the html template for badge earner page
- issuer.html : This is the html template for badge issuer page

- public_root/channel.html : This is the static content required by Facebook
- public_root/favicon.ico : This is the statuc content for icon use in the browser
- public_ver/app.js : This is the client java-script
- public_ver/style.css: This is the css file use
- still more.....

5. DATABASE DESIGN

5.1 MongoDB

MongoDB is a scalable, high-performance, open source, NoSQL document-based database. MongoDB features include document-oriented storage, indexes, replication, high availability, auto-sharding, and querying. [20]

5.2 MongoLab

MongoLab is the cloud computing provider for MongoDB database, easily integrated with Heroku. [21]

5.3 Documents

Data in MongoDB is stored in documents, and every document must have a primary key named _id. Like requlars SQL-based database, documents are like row in a table, where in MongoDB, documents have flexible schema, but every document must have _id field, even if you don't speficy the _id field, mongoDB will add that automatically. [20]

Although document structure is not enforced in MongoDB, different data model and structure may have significant impacts on MongoDB and application performace. So it is good to keep some kind of structure or pattern in data model.

5.4 Collections

Documents in MongoDB are organized in collection, and basic database operations are performed based on collection. Indexes can be assigned in any field or subfield contained in documents within a MongoDB collection, and they are defined on per-collection level. [20]

5.5 Data Model

In this project, there are three main collections, User, Group and Badge Colletion. And there are four linkings colletions to represent the many-to-many relationship between the main collections.

- user : this collection contains the user information
- group : this collection contains the group information
- badge : this collection contains the badge information
- group_admin_links : this linking collection contains group_id and user_id, which represent the admin of a group
- group_member_links : this linking collection contains group_id and user_id, which represent the member of a group
- badge_user_links : this linking colletion contains badge_id and user_id, which reperesnt badges that user earned
- group_badge_links : think linking collection contains group_id and badge_id, which represent badges that belong to a group

6. PROJECT IMPLEMENTATION

The GradeBadge application is designed to work on mobile, tablet devices and desktop computers. The UI of the application is developed using Bootstrap. When a page requires the data to be loaded from server or modified or deleted, a request is sent to the Web server over HTTPS. The requests are sent to the Web server using Ajax. For handling Ajax requests and responses, this application uses JQuery Ajax API. All UI components are dynamically created or initialized in response to the data received from the Web server.

6.1 Loading Screen

When the GradeBoard application is loaded, a loading screen is presented to the user as shown in the Figure 6.1. The loading screen shows application logo and loading progess bar, and the screen is automatically redirected after the loading completed.

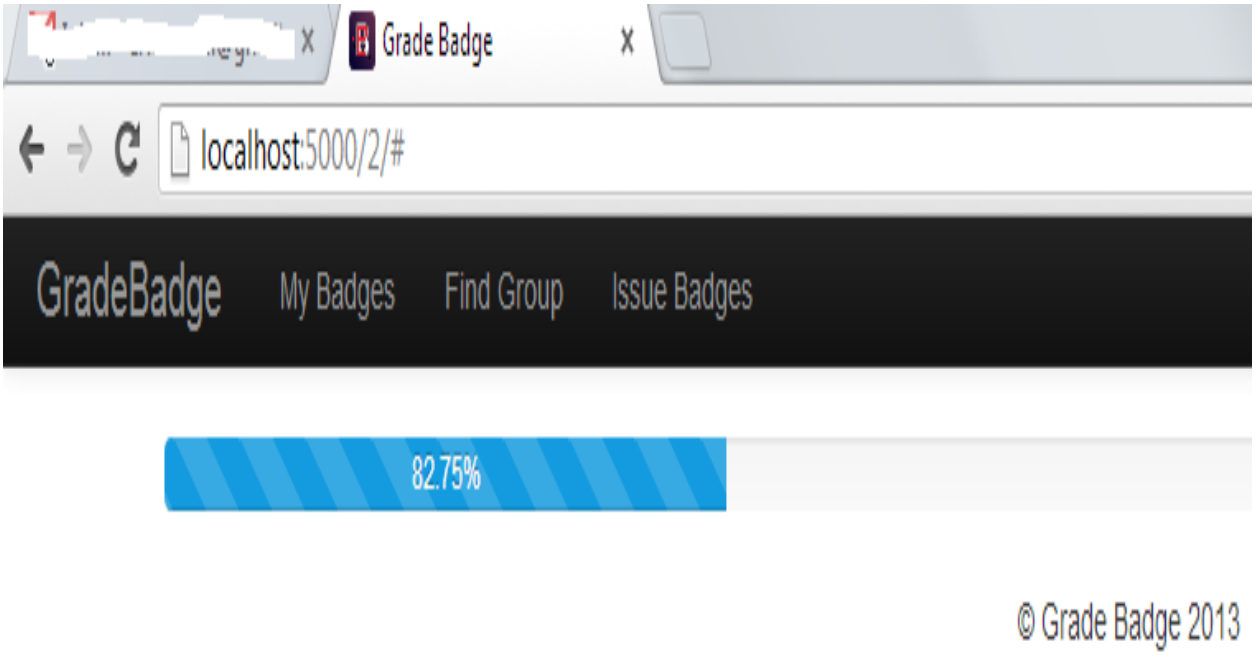

Fig. 6.1: GradeBadge Loading Screen

6.2 Login Screen

GradeBadge uses Facebook account for users to login. When the user is not logged-in to Facebook, the screen is automatically redirected to the Facebook login screen as shown in Figure 6.2. Every user in the system can be badge issuer and badge earner.

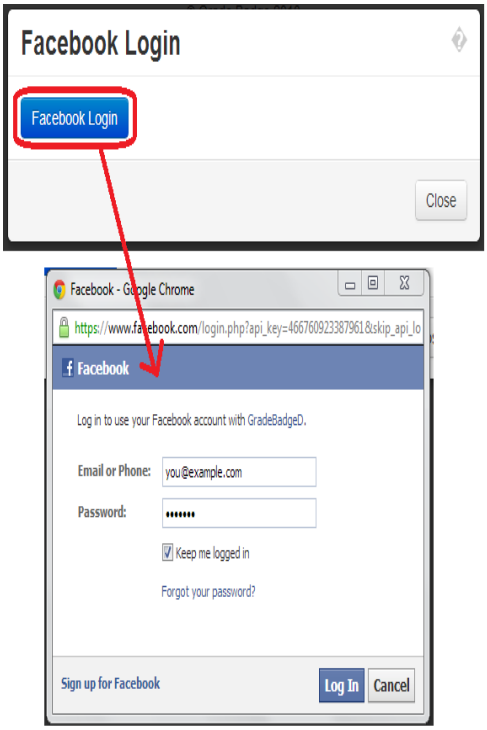

Fig. 6.2: GradeBadge Login Screen

7. CONCLUSION AND FUTURE DIRECTION

7.1 Conclusion

Conclusion here

7.2 Future Direction

Future Direction here

- Provide a responsive design using technologies such as Bootstrap to support dynamic screen sizes on multiple devices and desktop systems [4].
- item here
- Add new interfaces to the data model to support alternative database systems that could be used at lower cost, such as MongoDB [20].

APPENDIX A

SOURCE CODE

```
//main.js
```

REFERENCES

[1] Ajax (programming). http://en.wikipedia.org/wiki/Ajax_(programming).

[2] Amazon Web Services. http://aws.amazon.com/.

[3] Android Mobile Computing Platform. http://developer.android.com/about/index.html.

[4] Bootstrap. http://twitter.github.com/bootstrap/.

[5] Eclipse Fouccation. http://www.eclipse.org/.

[6] Entities, Properties, and Keys. https://developers.google.com/appengine/docs/java/datastore/entities.

[7] Facebook Developers. https://developers.facebook.com.

[8] Github. https://github.com/.

[9] Glossary. http://technet.microsoft.com/en-us/library/bb742416.aspx.

[10] Google Developers Academy. https://developers.google.com/appengine/training/intro/whatisgae/.

[11] Heroku How It Works. http://www.heroku.com/how.

[12] How Entities and Indexes are Stored. https://developers.google.com/appengine/articles/storage_breakdown.

[13] Http secure. http://en.wikipedia.org/wiki/HTTPS.

[14] Index Definition and Structure. https://developers.google.com/appengine/docs/python/datastore/indexes.

[15] IOS Mobile Computing Platform. http://en.wikipedia.org/wiki/IOS.

[16] Jetty (web server). http://en.wikipedia.org/wiki/Jetty_(web_server).

[17] Jquery Mobile 1.2 Reference Document. http://www.jquerymobile.com/demos/1.2.0/docs/about/features.html.

[18] Jquery Official Document. http://www.jquery.com/.

[19] Json. http://en.wikipedia.org/wiki/Json.

[20] Mongodb Agile and Scalable. http://www.mongodb.org/.

[21] Mongolab. http://www.mongodb.org/.

[22] Node.js Manual and Documentation. http://nodejs.org/api/.

[23] Nosql. http://en.wikipedia.org/wiki/NoSQL/.

[24] Official Documentation of Google App Engine Java Datastore. https://developers.google.com/appengine/docs/java/datastore/.

[25] Official Java Technology Document. http://www.java.com/en/download/faq/whatis_java.xml.

[26] PolyModel. https://developers.google.com/appengine/docs/python/ndb/polymodelclass.

[27] Unified Modeling Language. http://www-01.ibm.com/software/rational/uml/.

[28] W3C community. http://www.w3.org/TR/REC-html40/.

[29] What is an Object? http://docs.oracle.com/javase/tutorial/java/concepts/object.html.

[30] Windows Azure. http://www.windowsazure.com/en-us/develop/overview/.

[31] Manoj Kulkarni. *GradeBoard: A Cloud-Based Solution for a Student Grading System.* CSUSB - Masters Project, 2013.

[32] Ph.D Steve Burbeck. Mvc How to use Model-View-Controller. http://st-www.cs.illinois.edu/users/smarch/st-docs/mvc.html.