

# WeConnect

## Project Documentation

Seung H. Cha  
Chang-yong Choi  
Na Guo  
Jin Mo Ku  
Seungchul Lee  
Alexander Truong

University of Illinois at Urbana Champaign  
CS428 : Software Engineering II  
Prof. Darko Marinov  
Spring 2014

# Table of Contents

## [I. Description](#)

## [II. Process Followed](#)

- [1. Issues of iterative development](#)
- [2. Issues of refactoring](#)
- [3. Issues of testing](#)
- [4. Issues of collaborative development](#)

## [III. Requirements & Specifications](#)

- [1. User Stories](#)
- [2. User Cases](#)

## [IV. Architecture & Design](#)

- [1. UML Diagram](#)
  - [1.1 Class Diagram : Overall Structure](#)
  - [1.2 Class Diagram : Bookmark & Application of an Announcement\\*](#)
  - [1.3 Sequence Diagram:](#)  
[Sending a resume to the user who posted the announcement.](#)
- [2. Describe system architecture](#)
- [3. System Design](#)
- [4. Framework Choice and Influence](#)

## [V. Command to generate Code Documentation](#)


## [VI. Future Plans](#)

- [1. Personal reflections](#)

## I. Description

The goal of our project is to build a website where students and faculty members can share their interests through a simple networking service. Students and faculty members can form an academic network through which they can find each other's profiles, research needs or interests, and university events. The website will expose students to school events and research opportunities in which they are qualified to participate and enable professors to better advertise their projects. It also focuses on matching users with notifications that might intrigue them.

## II. Process Followed

We adopted Extreme Programming  with minor changes as a software development method

- Pair programming is recommended, but not required
- TDD is not strictly followed, so Spike, Testing then Coding

### 1. Issues of iterative development

- Learning a new platform and frameworks (Node.js, MongoDB, Mocha)
- Implementing appropriate databases and their schemes
- Implementing networking among users and reflecting it on databases.
- In ideal case, supporting multi-platforms (PC, Mac, Mobile)
- Make application has a dynamic schema and utilized them when needed.

### 2. Issues of refactoring

- By installing a server, the least infrastructure has been ready.
- Re-factored the application structure and design to better fit the callback natures of Node.js + MongoDB.
- We discovered that there were problems with session management: MongoDB bson types were incompatible with user variables stored in session as json types (problem solved).

- We discovered that we might have to change our DB access functions in order to more effectively fetch followees/ers list (we will be making this change next iteration for more effectiveness).
- Storing images as thumbnails
- Adjusting image size and viewpoint
- Responsive vs. unresponsive design
- Find a perfect font

### **3. Issues of testing**

- Because of asynchronous I/O in Node.js and MongoDB, it was hard to make tests at first, but its resolved with a big effort by Seung H. Cha and Seungchul Lee.
- Making tons of test cases for each user stories

### **4. Issues of collaborative development**

- Managing the project with newly incoming pairs smoothly
- Flexible Scheduling through GroupMe communication.
- Reading/Understanding other team members' code which are relate to the part that oneself is working on
- Conflict merge when work on multiple computer for the interface design

### III. Requirements & Specifications

#### 1. User Stories

- A. As a student, I can create my academic profile including a list of CS courses, GPA, areas of interests such as data mining or machine learning, and a short description on myself.
- B. As a student, I can choose to follow other users to receive their notifications or updates on research opportunities and school events.
- C. As a student, I get automatic recommendations on announcements.
- D. As a student, I can RSVP or apply (send a resume) to the user who published the announcement to notify that I want to participate in it.
- E. As a student, I can post an announcement under the administrator's permission.
- F. As a student, I can bookmark an announcement so that I can easily access it.
- G. As a faculty, I can post an announcement regarding researches or events and set specific requirements and conditions that need to be fulfilled in order for other users to RSVP or apply.
- H. As a faculty, I can advertise my research or event pages to other users (through the recommendation system).
- I. As a faculty, I can choose to follow other users to receive their announcements.
- J. As a students/faculty who posted an announcement, I can check RSVPs or receive resumes from users interested in my announcement.
- K. As a students/faculty, I can modify the list of my followers and followees.
- L. As a students/faculty, I can create an account with public and private information that is secured locally and over the network and login for service.
  - a. authentication
  - b. general user profile
  - c. student info / faculty info.
- M. As a student/faculty, I can modify my profile such as changing, overall gpa, or change the homepage url.
- N. As a student/faculty, I can upload media contents (pictures, resumes.. etc).

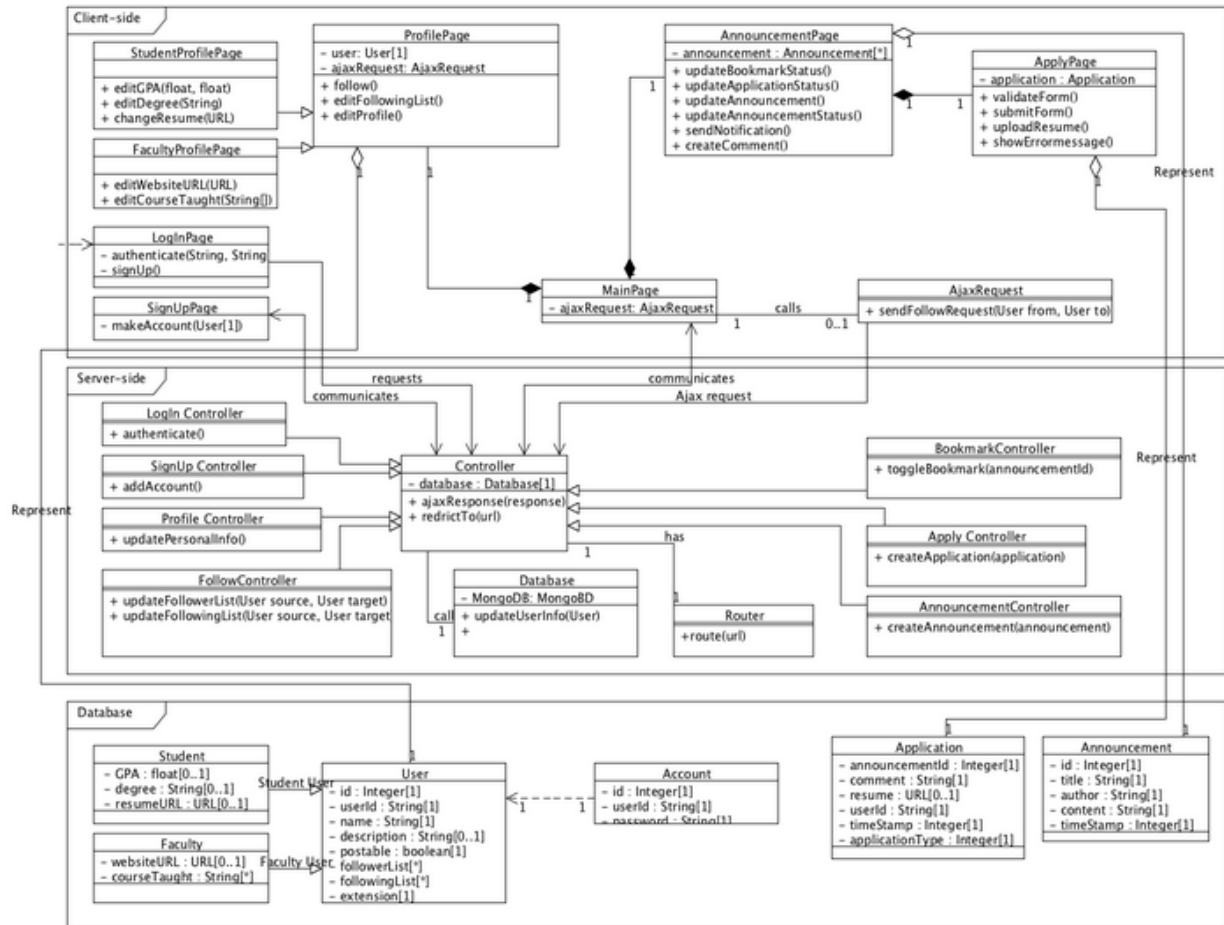
## 2. User Cases

Actor (users)	Task-Level Goal	Priority
Student, Faculty	Sign Up	1
Student, Faculty	Login	1
Student, Faculty	Follow other users	2
Student, Faculty	Edit the following list	2
Student, Faculty	Edit profile	1
Faculty	Post private/public announcements	2
Student	(Under the administrator's permission) Post private/public announcements	2
Administrator	Give permissions to students to post announcements	2
Administrator	Give approvals on pending announcements posted by students	2
Student	Bookmark an announcement page	3
Student	RSVP to an announcement	3
Student	Send a resume to the user who posted the announcement	3
Recommendation System	Recommend users to other users	2
Recommendation System	Recommend announcements to users	2
App	Provide views for announcements in the main page	2

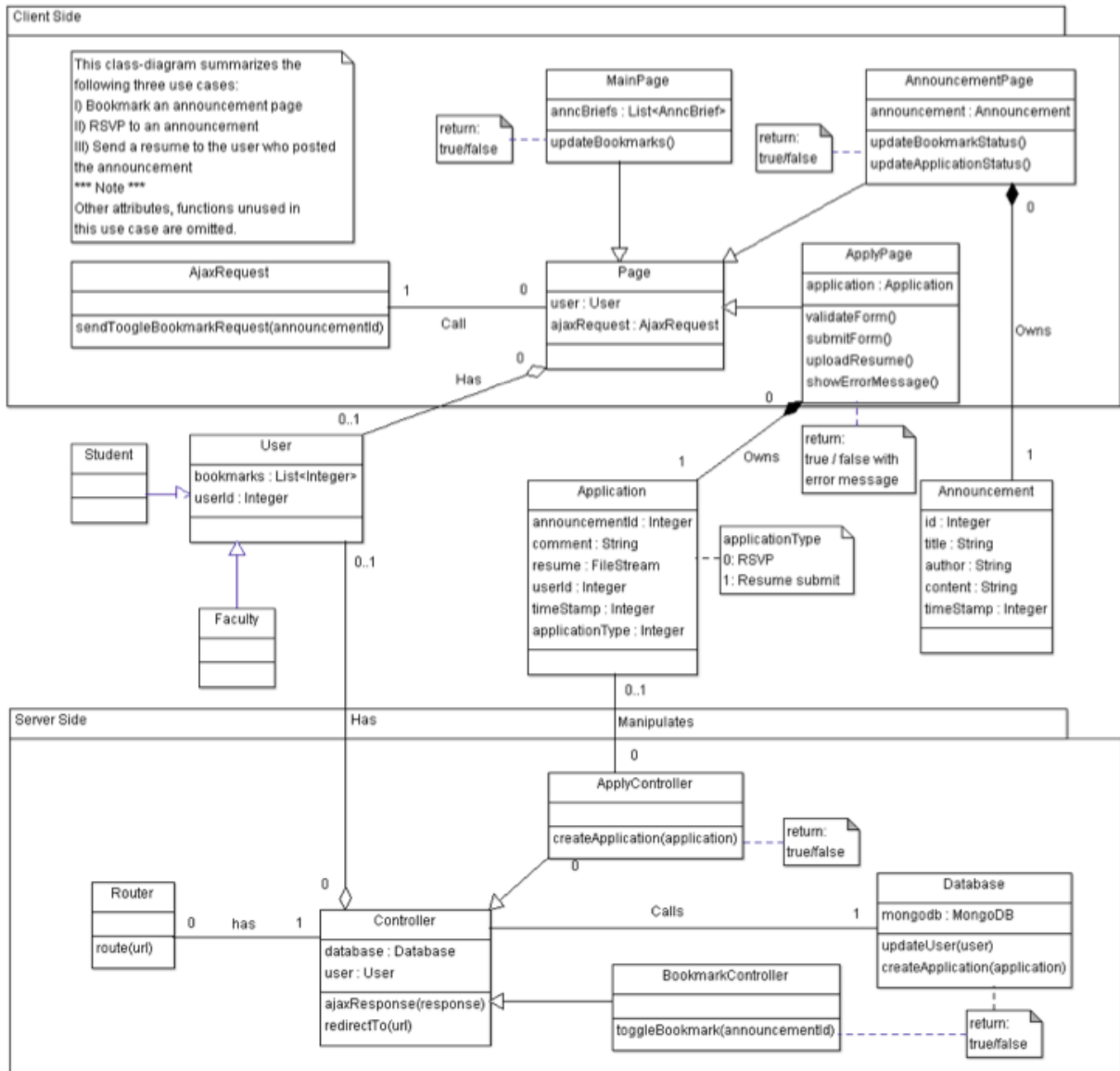
## IV. Architecture & Design

### 1. UML Diagram

#### 1.1 Class Diagram : Overall Structure



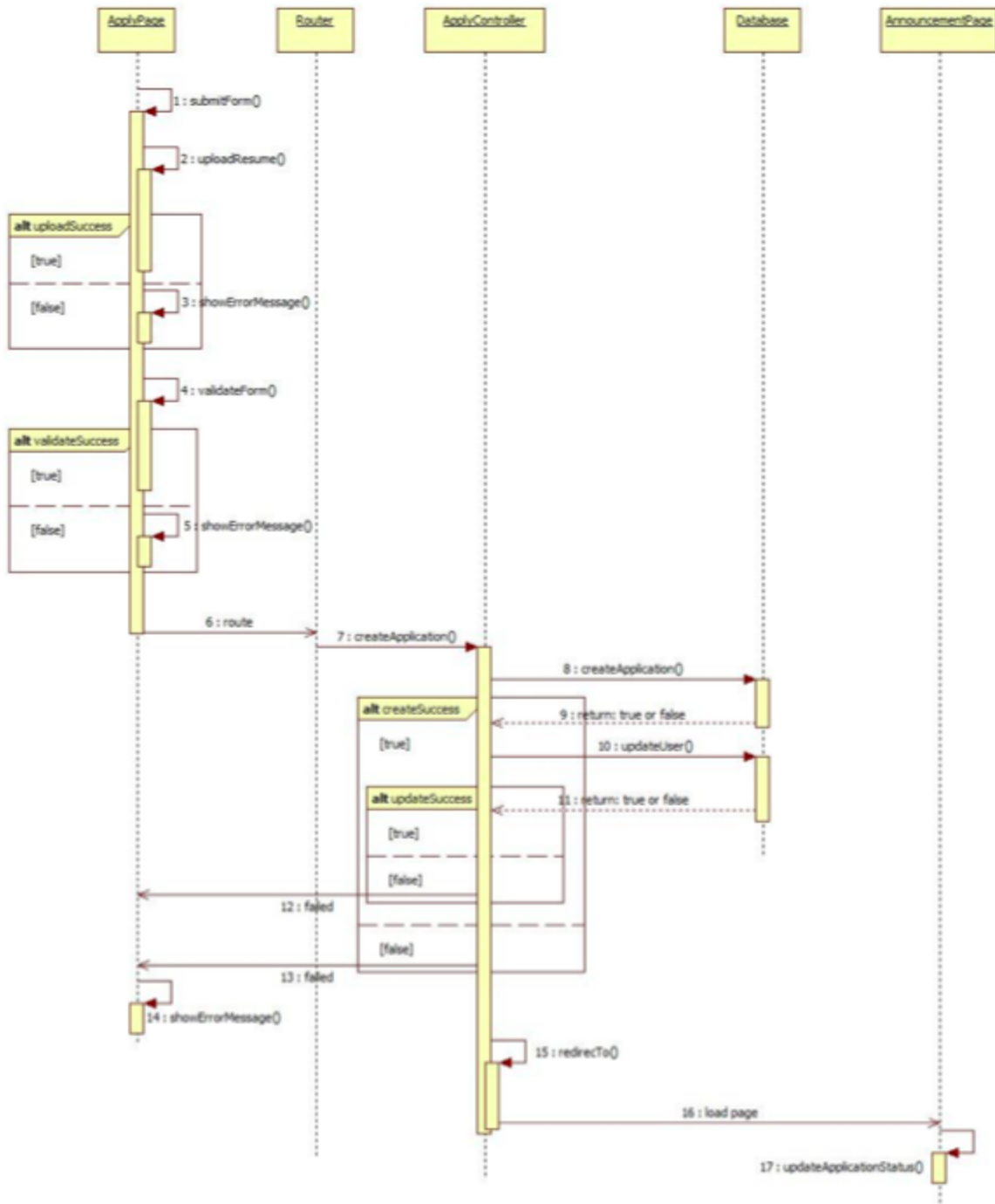
## 1.2 Class Diagram : Bookmark & Application of an Announcement





### 1.3 Sequence Diagram:

Sending a resume to the user who posted the announcement.



## **2. Describe system architecture**

The application can be seen as three parts in top-down manner. It has front-end view, router, and the backend controller with database access. Front-end view renders the data returned by the backend. Router plays as a middle layer that handles the URL request both static and dynamic ajax request directing the operation to the backend controllers. Backend controllers, when directed by the router, does appropriate database operation, and return back the queried data to the front-end views. Backend modules are consist of database module, controllers, and session manager.

## **3. System Design**

There are two major data objects in the application: user and announcement. Likewise, in the backend, there are two controllers, user controller and announcement controller, and they handles the related operation respectively.

Front-end view is divided into several modules: account, announcement, embedded, profile. Account and profile are for user data objects, and they are separated to enhance the security by separating secured part of user data to account. Announcement is for rendering announcement related data. Embedded is for ajax rendering which doesn't need common header and footer when rendering.

Front-end view also uses many of jQuery and Bootstrap features to enhance the view-ability and dynamic operations with less efforts.

Announcement and user recommended system of the application is embedded with the two database modules: user and announcement. They are designed to call basic and sophisticated MongoDB queries and get the necessary data efficiently.

User session is managed under session module and it handles sign in, sign out, and persistent data for signed in user to reduce the database call throughout the session.

#### **4. Framework Choice and Influence**

For the backend we used NodeJS and MongoDB as our primary development platform. NodeJS enables using Javascript for both front-end and backend operations. MongoDB enables us to make efficient dynamic queries per page.

The backend also make use of ExpressJS and NodeJS framework for efficient routing and session management. It also helped us decomposing the backend according to the functionality of each module. Along with ExpressJS we also used easyimage and nodemailer module which handles the image resizing.

For the backend-frontend data rendering, we used EJS template engine which is supported by the ExpressJS framework. This EJS template engine helps us effortlessly render queried data to the frontend views. Frontend views make use of jQuery library and Bootstrap CSS framework. jQuery enables us to easily perform XHR ajax request to the backend. Bootstrap helped us doing fast prototyping and designing. Using many Javascript features of Bootstrap framework also helped us reducing workload for modal dialog implementations.

## V. Command to generate Code Documentation

We are using the JSDoc API to auto-generate our code documentation. In order to auto-generate the documentation, follow the following steps:

Install JSDoc globally by running the following command in the terminal (this assumes Node.js has already been installed):

```
npm -g install jsdoc
```

After installing successfully, we run the following command to generate the document:

```
jsdoc -c conf.json
```

Another folder named **out** should have been created. We can navigate into the **out** folder and open the index.html file in our browser. We are then able to see the auto-generated documentation for the code.

## VI. Future Plans

For our future plan, we intend to implement a mobile application for smartphones which will work across all platforms and possible extensions for tablets.

### 1. Personal reflections

#### **Seung H. Cha**

Working on this project helped me learn a lot about new trending technologies, namely NodeJS and MongoDB. Although I had many development experience from previous jobs and classes, learning and actually working on the project with platform and non-relational database never used before was somewhat challenging. Asynchronous database access and unfamiliar data type like BSON was especially interesting. Throughout the project, I also learned the importance of documentation and communications among group members to make everyone is on the same track. Overall it was fun and meaningful project.

#### **Chang-yong Choi**

When we decided to select this project with some cutting-edge technologies ME(A)N Stack (mongodb, express, angular.js and node.js), I felt little excited that I could learn something new. Of course, it was fun in fact. however, it sometimes overwhelmed us and fell us down hard. It did give us such true challenges never easily overcome during the short term. At this point when the project is almost about to finish, now I understand how the routing works, what protocol in HTTP is used and how the web pages are rendered. Because of these achievements, I don't regret to choose this project.

#### **Na Guo**

Since we used many things that we were not familiar with previously, this project allows us to venture various new experiences. In other classes, we did not have similar opportunities to do something we proposed with a specific programming process, and we did not have chances to work in a large group to develop further coding skills from each other.

**Jin Mo Ku**

The project was a little overwhelming to me at first because this was the first time I ever planned a six-iteration-long project. However, my teammates were able to help me understand our extreme programming process. We divided our group as pairs to work on different sections our application. Because each of us always fulfilled our responsibilities, we were able to meet our project requirements on time. This project provided a great opportunity to learn node.js and express.js framework along with many important basic concepts and methods that are used in web development. I used NoSQL for the first time which was quite a new experience. I hope to work on more web app projects in the future.

**Seungchul Lee**

From making this web application, I had really good learning experiences. At the beginning of the semester, Me and my pair made a good proposal, but could not attract other people. Here, I learned how to attract other developers into my project. After joining my current team, we implemented and learned some different software development processing, such that extreme programming (XP), the Spike Solution and Test Driven Development in a hybrid manner that allows us to write efficient code as a student programming. Moreover, we used frameworks, such as Node.js, Express, and MongoDB that we had not used at all before. This experience teaches me how to learn new technologies in the real life.

**Alexander Truong**

Creating this application led to new learning experiences. Learning different frameworks, such as Express, and new languages, such as Node.js, required a lot of reading and understanding of advanced concepts. Following the process was tedious, but allowed for a streamlined way to do things which resulted in a strong scaffolding for future implementations. Although there were multiple issues both in group dynamics and the technical aspect, everything was resolved by fair debate. Working on the front-end really showed how difficult it was to things aesthetically appealing and function correctly.