

# Cover Sheet

CS5200  
Fall 2017  
Section 2  
Database Management Systems

Team 10 Member:

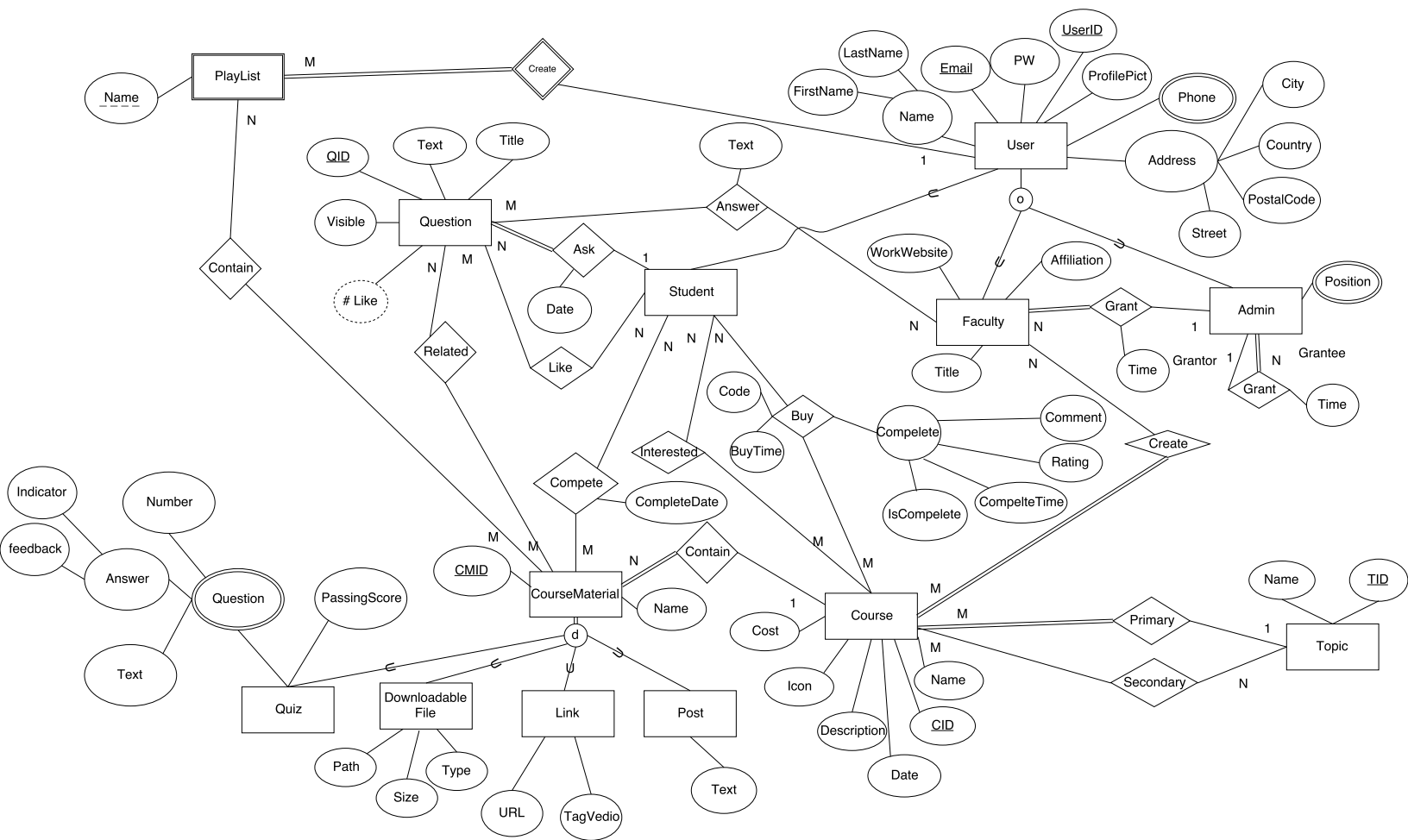
**Jiangyi Lin**

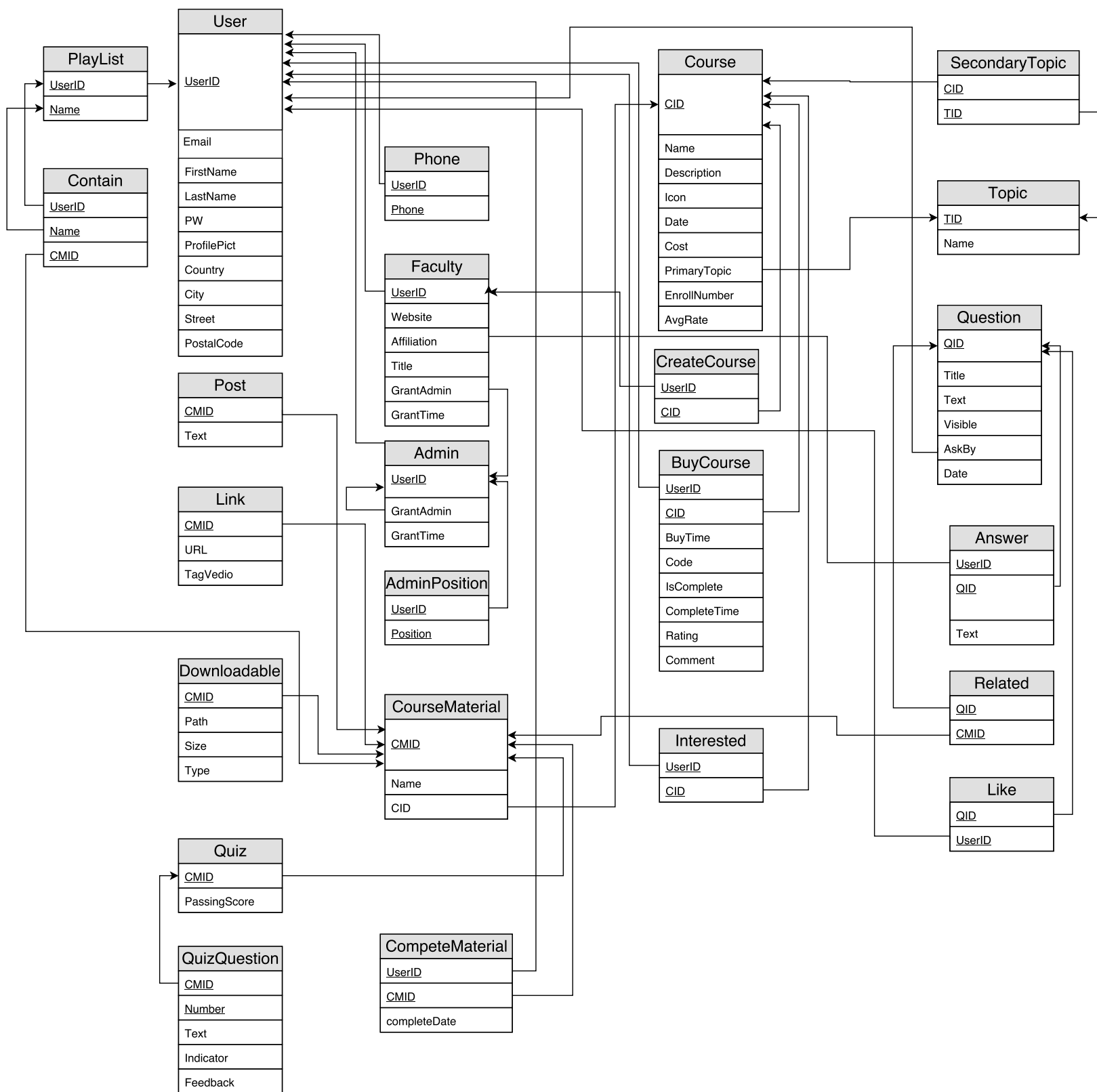
**Ni Ke**

**Wandong Wu**

**Yinxiang Wang**

Completion Date: Dec. 8, 2017





After following the README, you can already reach the main page of trainly.io.

http://127.0.0.1:8000/trainly

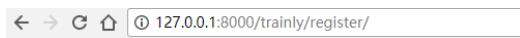


## Welcome to Trainly.io!

- [Log In](#)
- [Sign Up](#)
- [Logout](#)

The Trainly.io is an App Store of training where organizations with faculty can create and teach courses for users who find and enroll in.

You can register a new user.



## Register Now!

First Name:

Last Name:

Country:

City:

Street:

Postal Code:

Password:

Email:

The user's password will be added salt and database only store MD5.

We use their email address as the salt. One email can only register once, so the salt is also unique.

← → ↻ 🏠 ⓘ 127.0.0.1:8000/admin/trainly/user/52/change/

Home · Trainly · Users · 52:happy.lin.jiangyi@gmail.com

### Change user

Email:	<input type="text" value="happy.lin.jiangyi@gmail.com"/>
FirstName:	<input type="text" value="Jiangyi"/>
LastName:	<input type="text" value="Lin"/>
Pw:	<input type="text" value="861e76a35a256c11f1ae5e07c12a2859"/>
ProfilePict:	<input type="text" value="default.jpg"/>
Country:	<input type="text" value="USA"/>
City:	<input type="text" value="Boston"/>
Street:	<input type="text" value="123 St."/>
PostalCode:	<input type="text" value="12345"/>

You can login now.

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/login/

# Welcome back!

Email:

Password:

Only with same email and passcode you can successfully login.

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/ 🔍 ⚙️ ☆ ⓘ

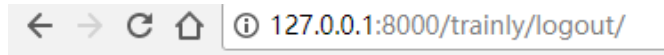
## Welcome to Trainly.io!

Hello, Jiangyi  
[main page](#)

- [Log In](#)
- [Sign Up](#)
- [Logout](#)

The Trainly.io is an App Store of training where organizations with faculty can create and teach courses for users who find and enroll in.

You can logout.



You have already logout

Now go to the main page of this new user.



Hello, Jiangyi Lin

## Basic Information



**Email:** happy.lin.jiangyi@gmail.com

**Country:** USA **City:** Boston

**Street:** 123 St. **PostalCode:** 12345

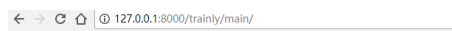
[See All Courses](#)

[Enrolled Courses](#)

[Completed Courses](#)

[Interested Courses](#)

The default user is not a faculty or admin. Let's look at another user.



Hello, Reginald Hearst

## Basic Information



**Email:** osd@tom.com

**Country:** USA **City:** Denver

**Street:** 2744 Davis Lane **PostalCode:** 80202

[See All Courses](#)

[Enrolled Courses](#)

[Completed Courses](#)

[Interested Courses](#)

## Admin Site

[Add a faculty](#)

[Add a fellow admin](#)

## Teacher Site

**affiliation:** Mathematics

**website:** [www.wiley.com](http://www.wiley.com)

**title:** Academic Professor

If he/she is a faculty or admin, he/she will get the content below.

You can add a user as faculty, but it must exist.

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/add\_faculty/

## Write faculty's Email

Email:

Website:

Affiliation:

Title:

Or you can only get there is no such user.

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/add\_faculty/

## User not exist

Give the new user faculty privilege.

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/add\_faculty/

## Write faculty's Email

Email:

Website:

Affiliation:

Title:

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/add\_faculty/

Done!

It is the same thing for admin

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/add\_admin/

# Write admin's Email

Email:

Submit

Wrong email.

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/add\_admin/

User not exist

Right email.

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/add\_admin/

Done!

Adding twice is unnecessary.

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/add\_admin/

This user has been a Admin


Now this new user has been faculty and admin



← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/main/

Hello, Jiangyi Lin

## Basic Information



**Email:** happy.lin.jiangyi@gmail.com  
**Country:** USA **City:** Boston  
**Street:** 123 St. **PostalCode:** 12345  
[See All Courses](#)  
[Enrolled Courses](#)  
[Completed Courses](#)  
[Interested Courses](#)

## Admin Site

[Add a faculty](#)  
[Add a fellow admin](#)

## Teacher Site

**affiliation:** Computer Science  
**website:** [www.example.com](#)  
**title:** Lecturer

You can see all courses and add them.

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/courses/

- [1:Calculus 1](#) [Add this Course!](#)
- [2:Calculus 2](#) [Add this Course!](#)
- [3:Calculus 3](#) [Add this Course!](#)
- [4:Fundamentals of Computer Science 1](#) [Add this Course!](#)
- [5:Fundamentals of Computer Science 2](#) [Add this Course!](#)
- [6:Theory of Computer Science](#) [Add this Course!](#)
- [7:Education in the Community](#) [Add this Course!](#)
- [8:Game Artificial Intelligence](#) [Add this Course!](#)
- [9:Fndtns Artificial Intellegence](#) [Add this Course!](#)
- [10:General Physics](#) [Add this Course!](#)
- [11:Physics for Life Sciences](#) [Add this Course!](#)
- [12:Transport Processes](#) [Add this Course!](#)
- [13:Chemical Engr Process Control](#) [Add this Course!](#)
- [14:Machine Learning](#) [Add this Course!](#)
- [15:Principles of Macroeconomics](#) [Add this Course!](#)
- [16:Principles of Microeconomics](#) [Add this Course!](#)
- [17:Mathematical Statistics](#) [Add this Course!](#)
- [18:History of the United States](#) [Add this Course!](#)
- [19:Historical Research & Writing](#) [Add this Course!](#)
- [20:Topics in Cultural History](#) [Add this Course!](#)

At first, your enrolled, completed or interested list maybe empty.

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/enrolled\_courses/52

This user is enrolled in no courses.

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/completed\_courses/52

For this user, no Courses are completed.

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/interested\_courses/52

This user is interested in no courses.

Then you can add courses.

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/add\_course/3

Done!

But you can't buy it twice.

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/add\_course/3

You have already bought it!

Then if the list is not empty.

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/enrolled\_courses/1

- [1:Calculus 1](#)
- [2:Calculus 2](#)
- [3:Calculus 3](#)

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/completed\_courses/1

- [1:Calculus 1](#) [Get Your certification.](#)
- [2:Calculus 2](#) [Get Your certification.](#)

You can print your certification as long as you finish them!

← → ↻ 🏠 ⓘ 127.0.0.1:8000/trainly/interested\_courses/1

- [1:Calculus 1](#)
- [2:Calculus 2](#)
- [3:Calculus 3](#)

You can go to each course main page to see its topic and description

1:Calculus 1

MATH

Price: 800

Main Topic: Differential and Integral

Secondary Topic:

Undergraduate study ; Basic Mathematics study ;

## Materials

- Syllabus for Calculus 1 [Learn the material!](#)
- Calculus for beginners [Learn the material!](#)
- Introduction to learning Calculus [Learn the material!](#)
- Quiz 1 [Learn the material!](#)
- Quiz 2 [Learn the material!](#)
- Quiz 3 [Learn the material!](#)

But if you are not enrolled, you can't see the material.

11:Physics for Life Sciences

PHYS

Price: 900

Main Topic: life and health science

Secondary Topic:

Undergraduate study ;

Now learn the material.

You learn the material, you can still learn this course!

You can learn for several times.

You learn it again, good!

And when you finish all materials, you also finish the course!

You learn the material, you also finish the course!

You can get your certification now!

- [1:Calculus 1](#) [Get Your certification.](#)
- [2:Calculus 2](#) [Get Your certification.](#)
- [3:Calculus 3](#) [Get Your certification.](#)

You can print your certification as long as you finish them!

## Certification

Student **Reginald Hearst** finished Calculus 3 in Dec., 07, 2017

Trainly.io

## Project Retrospective

In this project, the most meaningful thing is that I do a web program from zero to a although primitive but workable project. I learned Django myself, following its tutorial. The tutorial goes from code to a database DML, but I need to generate model code from DML. So, I must also find more information from the Internet. And they are all very basic, common and useful skills. Using Django can build a website fast, and it has an admin site which can see data in database conveniently. MVC model is very clear and easy to use, url pattern is helpful, and HTML template is not a hard thing to learn. I really like Django and want to also continue to learn this skill after the course. The CSS and JS are hard, I don't mean I dislike them, but they need to be learnt in the future.

Good tools help me a lot, but it is more important to learn some principles inside, or I will meet many pitfalls. It's easy to make a web page and manage the hyper links based on Django. However, when problem came out, sometimes it's hard to debug. The information of exception is kind of not directly and difficult to understand. So, write it right at first and understand what I want to do first is very important.

I don't think it's a large-scale project, because I still think it's primitive and I will modify it after the course. There are so many good books for Django and very useful communities to find help. And modifying the old program will be definitely suffering for maybe many things I do will find a better way, I think I will learn how to start a good code and modifying it. Of course, more data will cause performance problem which I must handle.

## Conclusion

All in all, I learnt a lot in this course, as well as the project. Although what I did is still primitive, it is a website anyway. This website also gets and changes information in database, so I have learnt the most useful parts of web development and how to use a database. It completes all tasks professor requires but there are still many things to do.

First, I will want to learn HTML, CSS and Javascript to make website more beautiful. It will surely improve the quality and first impression when someone see this project. Second, Django does not support composite keys well, and should use a surrogate key. But I find this issue at half of my project, it's hard to make changes at that time. The same thing is for the salt, in ERD, there is no attribute for salt. I use email as a salt, but yes, this schema still has space for modification. I can only find these problems when I write the code, and I hope to make more beautiful and better project in the future.