CSCE 606 Software Engineering
Final Project Report

Team SOS: Yu Cai, Zhenyang Gu, Fuhao Shi, Junjie Shen, Xiangzhou Xia, Peizhao Zhang

Product: www.tamuchineseschool.com

1. Project summary

We have built a website with customer-specified databases for TAMU Chinese School. Their current website is built by google site, and the changes of course and teacher information need to be updated manually. The goal of this project is to provide a new website that has a new view, and provides efficient information management and automatic page update.

We have implemented the following to meet the customer needs:

1) Several databases established, including user/teacher/administrator management; course management; event management, etc.
2) A course registration/drop system that each family can choose which of their children when registering a particular class.
3) Contact page that parent/future parent can send message to the right person they want to contact.
4) Volunteer event management.
5) Administrator query. Administrator can login and query every user's (including parents) contact information. The stakeholders are administrators/teachers and parents of the Chinese school.

2. User stories

The user stories and other information are summarized here.

User Story	Points	Status	Notes (changes)
1. Add a class to the database	2	Finished	
2. Register a student to a class	2	Finished	
3. Add a teacher to the database	2	Finished	Implemented as user role changes
4. Student information query	2	Finished	
5. Contact to an administrator/teacher	2	Finished	
6. Volunteer sign up	3	Half done	Events management finished. Sign up onsite for now. Agreed by customer.
7. English/Chinese view change	1	Not started	Agreed by customer

Besides, proper user authentication control has been added.

As communicated with the customer, we have focused on the core functionalities, which are class/user/registration database and management. These parts have been fully implemented and the main needs of customer (efficient and automatic info management) are successfully achieved.

3. Team role

Product Owner	Fuhao Shi
Scrum Master	Xiaxiang Zhou

There is no role change.

4. Scrum iteration schedule

Start date	Description	User Story	Points to date
11.4, 2014	Lo-fi UI and Storyboards finalized. Database design starts.		
11.18, 2014	Setting up ROR environment. Initial database finished.	1,3	4
12.6, 2014	Database refinements and tests. Page designs.	2,4,6	10
12.12, 2014	Database-Page Connected. User-authentication added. Final testing.	5	13

5. Customer meeting dates

Meeting dates	Description
10.26, 1:30pm	Gathering needs from customer. Show the lo-fi UI.
11.9, 1:30pm	Clarifying item definition in the database. Show the initial database design
11.23, 1:30pm	Show the page design. Adjust plan to focus on database design. Add other pages as many as possible.
12.14, 1:30pm	Demo the final website. Gathering comments for refinement.

6. BDD/TDD process

We have used cucumber and RSpec for BDD/TDD testing. Our BDD uses the user stories and lo-fi UI to describe the features and the behaviors of our website before the implementation. It is easily to reach an agreement with customers before writing the code and avoid many unnecessary changes. However, it has the weakness that it is hard to see all the website architectures clearly. For the TDD, we first made a simple homepage with basic functions and let it pass our preset tests. Then, we added another feature.

We have found redirection errors and other exceptional cases (incorrect view/controller) through RSpec. The cucumber testing enables us to work easily with the non-technical customer, and makes sure we have implemented what the customer needs.

7. Configuration management

We have 68 commits, 1 branch and 1 release in total. Generally, we use Git for collaboration of the database development. The main challenge here is to get familiar with ROR framework, since the resources online are not quite up-to-date. We sometimes spend hours to search for a solution for a single feature, and make trial changes to the database. All these are committed in the process.

8. Issues in the production release process

Once we tried to release our production, we adjust one of the layout of our website, then we found all the other layout of our websites were changed as well. So we spend several hours to find a way to reload the layout for each website. Finally, we added the layout file in the precompile stage and fixed the problem.

9. Implementation environment

We put our project on a web server of cloudapp.net, so we can let our customers to see our website conveniently. Some of our members built their rails on Linux/Mac OSX systems. Some of them use windows system. The windows system made more troubles than the Linux system especially when we need to set the environment configuration. So instead, a VM running Ubuntu is actually used. We use Rubymine IDE for development.

10. Other tools we used

We use Git to do the version control for our project. This tool helps us to record the changes and modifications of our project. We can easily update our local files to the latest version by using Git. We use Git basically for development collaborations, as stated in 7. It helps us to check or return the status of our project to a certain point. It is very useful when some unexpected mistakes occur.

11. Link to customer interview:

http://youtu.be/Tek3lC4Tpi4

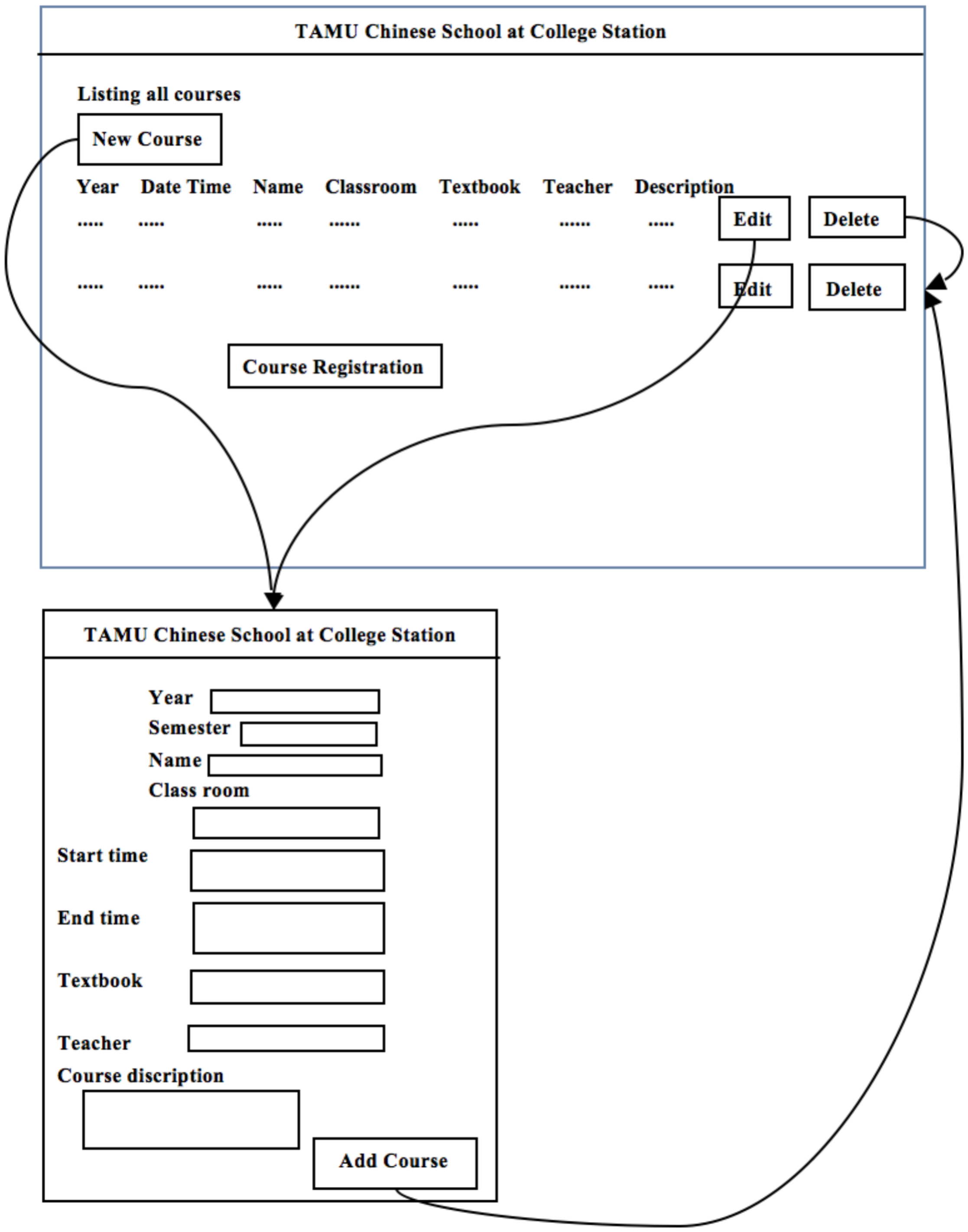
12. Link to final demo:

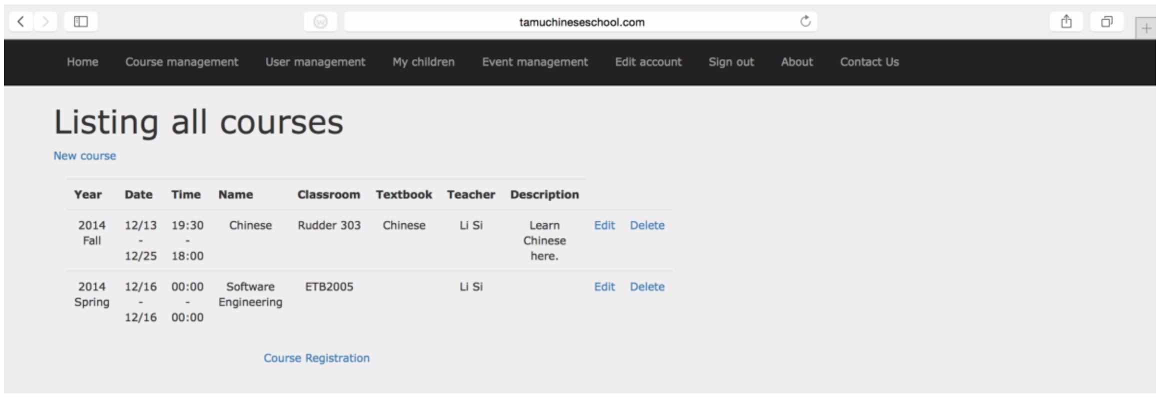
http://youtu.be/jUEQfJM77No

Snapshot

- User story storyboards and pages (corresponding to user stories in 2). It can be best seen in our final demo video.

User Story: Add a class to the database





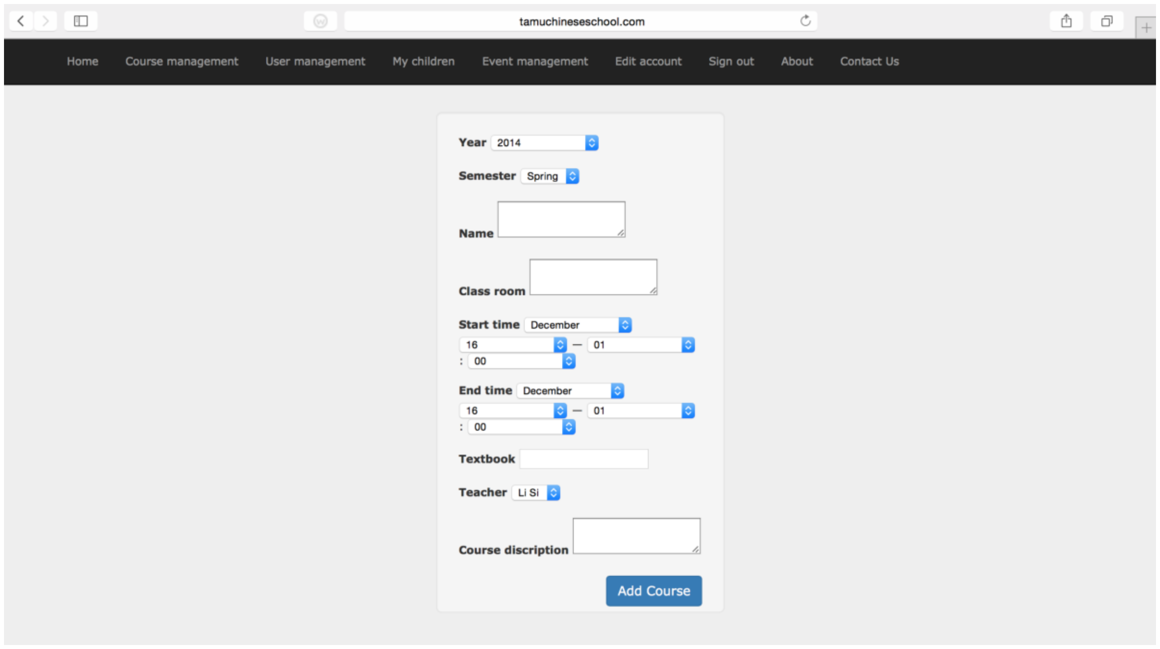
tamuchineseschool.com

Listing all courses

New course

Year	Date	Time	Name	Classroom	Textbook	Teacher	Description		
2014 Fall	12/13 - 12/25	19:30 - 18:00	Chinese	Rudder 303	Chinese	Li Si	Learn Chinese here.	Edit	Delete
2014 Spring	12/16 - 12/16	00:00 - 00:00	Software Engineering	ETB2005		Li Si		Edit	Delete

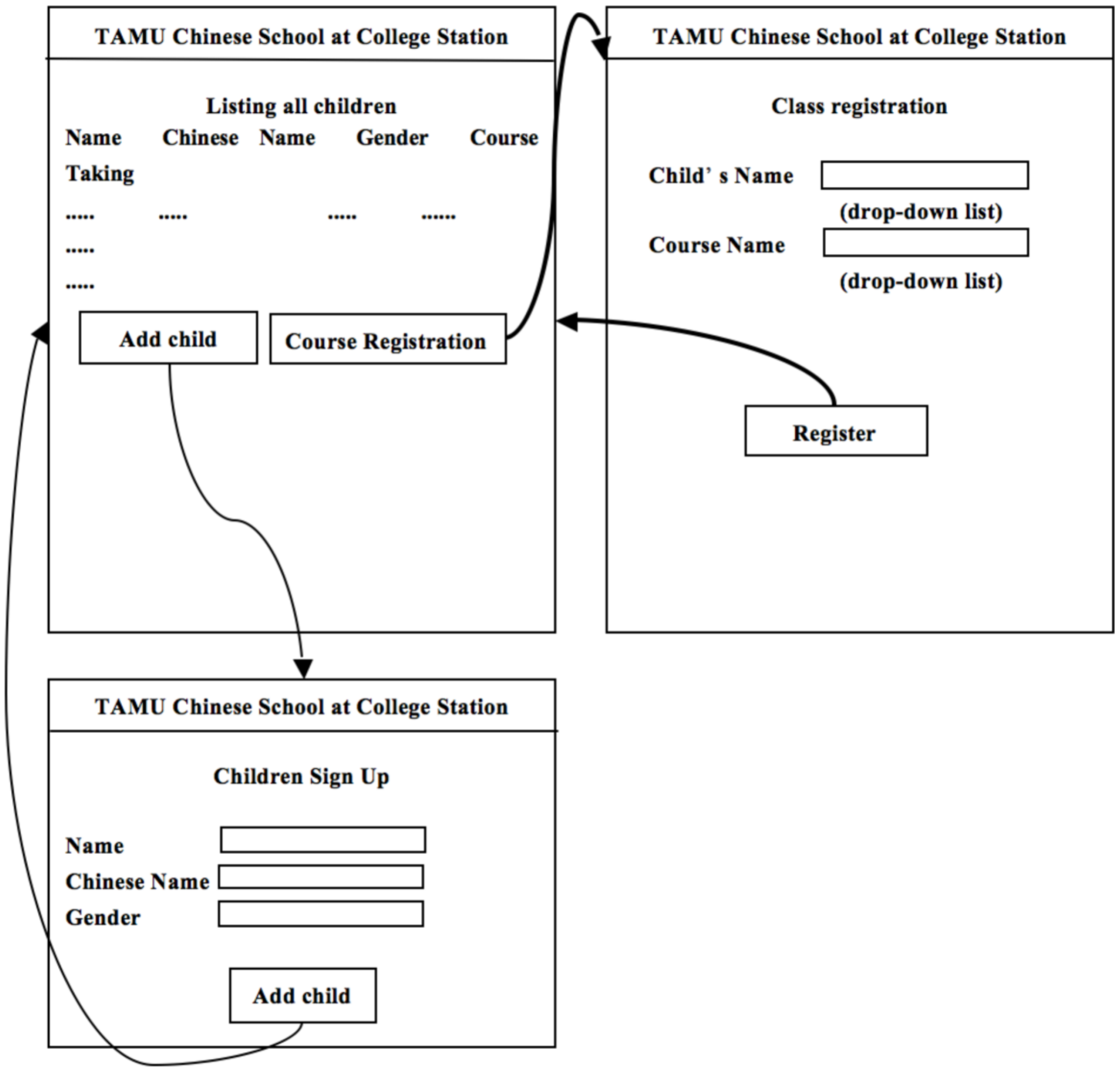
Course Registration

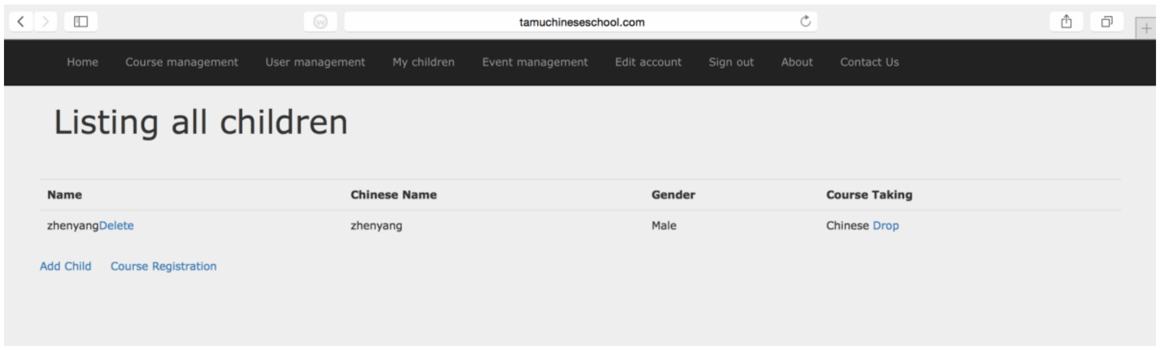


tamuchineseschool.com

Year	2014
Semester	Spring
Name	
Class room	
Start time	December 16 — 01 : 00
End time	December 16 — 01 : 00
Textbook	
Teacher	Li Si
Course discription	

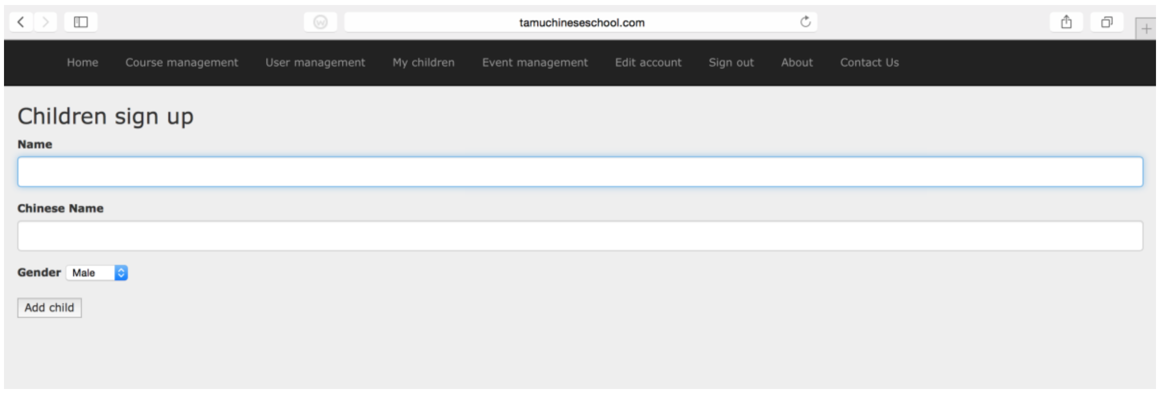
User Story: Register a student to a class





Listing all children

Name	Chinese Name	Gender	Course Taking
zhenyangDelete	zhenyang	Male	Chinese Drop


tamuchineseschool.com

Home | Course management | User management | My children | Event management | Edit account | Sign out | About | Contact Us

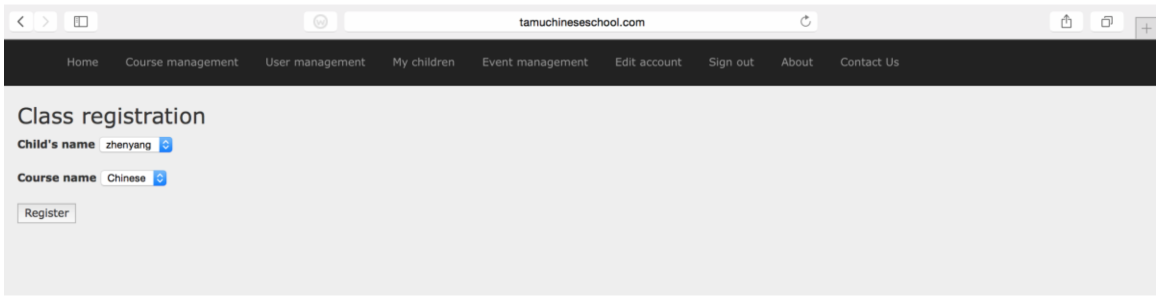
Children sign up

Name

Chinese Name

Gender Male

Add child


tamuchineseschool.com

Home | Course management | User management | My children | Event management | Edit account | Sign out | About | Contact Us

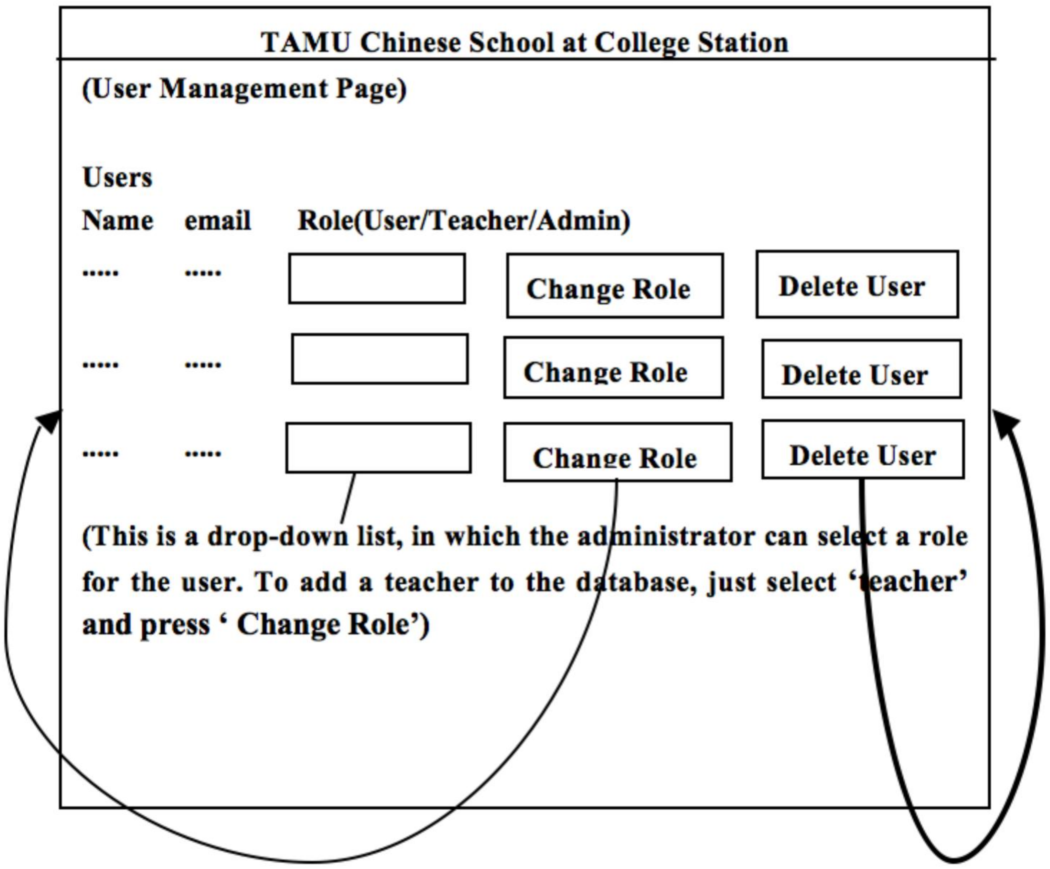
Class registration

Child's name zhenyang

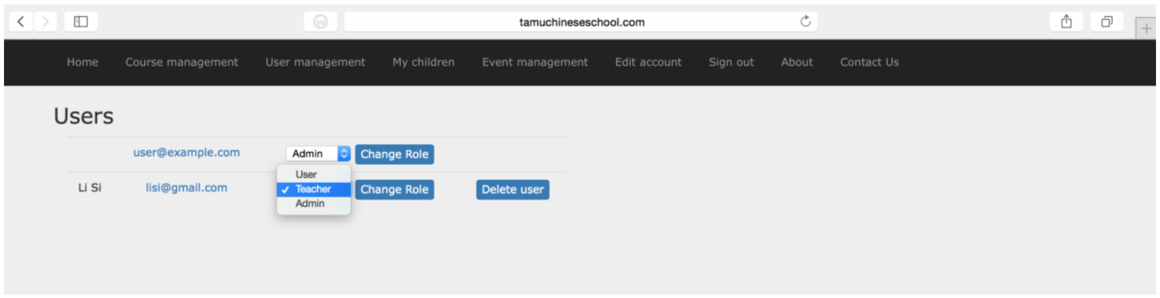
Course name Chinese

Register

User Story: Add a teacher to the database



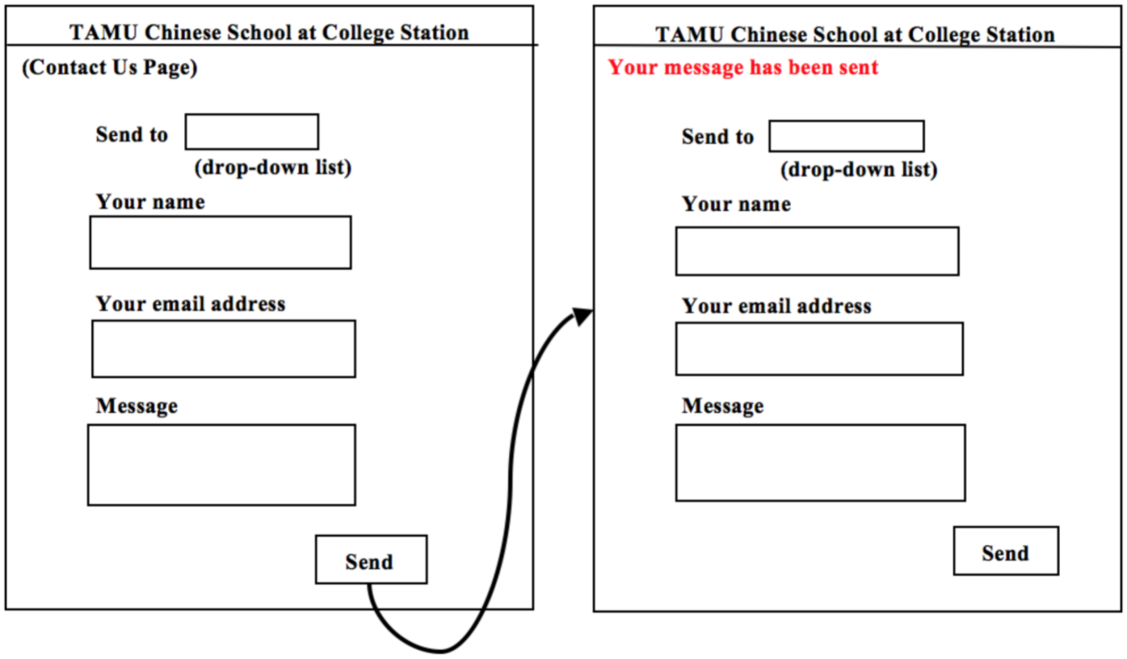
(This is a drop-down list, in which the administrator can select a role for the user. To add a teacher to the database, just select 'teacher' and press ' Change Role')

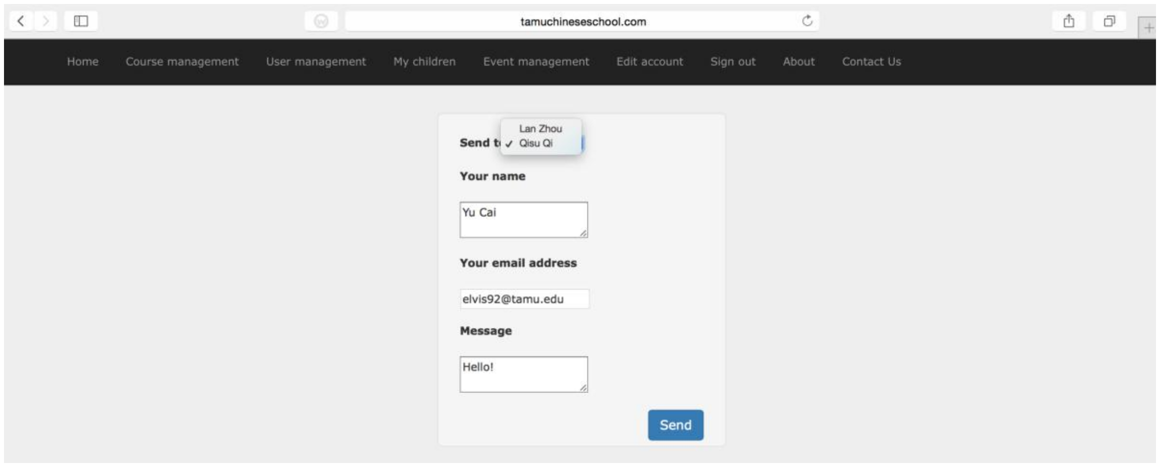


Users

	user@example.com	Admin	Change Role	
Li Si	lisi@gmail.com	User / ✓ Teacher / Admin	Change Role	Delete user

User Story: Contact to an administrator/teacher



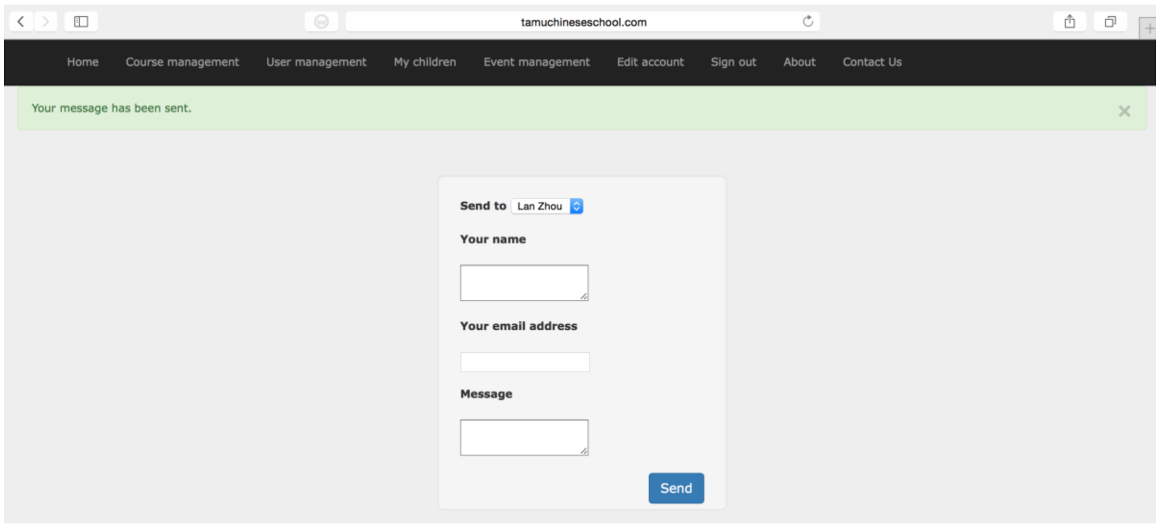


Send to: [✓ Qisu Qi]

Your name: Yu Cai

Your email address: elvis92@tamu.edu

Message: Hello!



Your message has been sent.

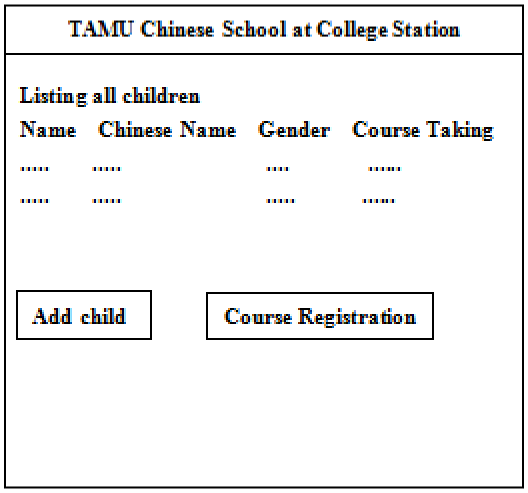
Send to Lan Zhou

Your name

Your email address

Message

User Story: Student Information Query



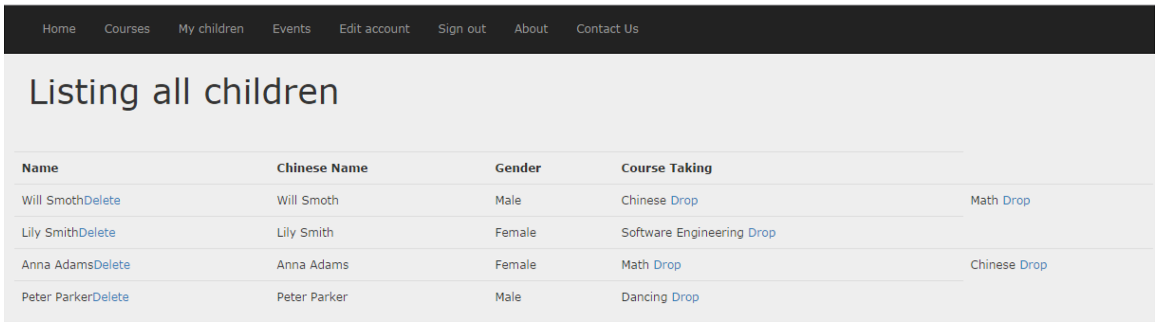
TAMU Chinese School at College Station

Listing all children

Name	Chinese Name	Gender	Course Taking
.....	.....	....	......
.....	.....	.....	......

Add child

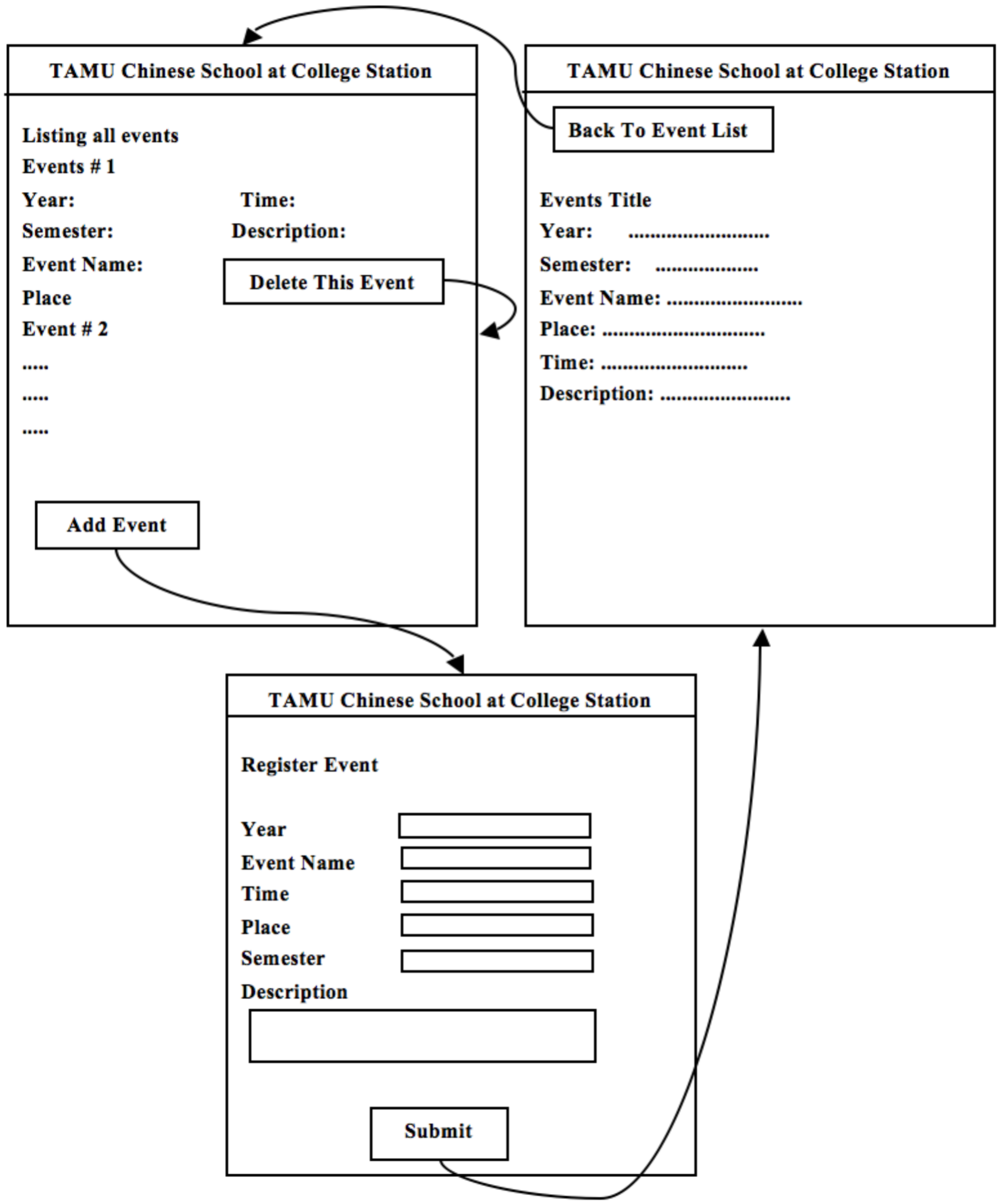
Course Registration

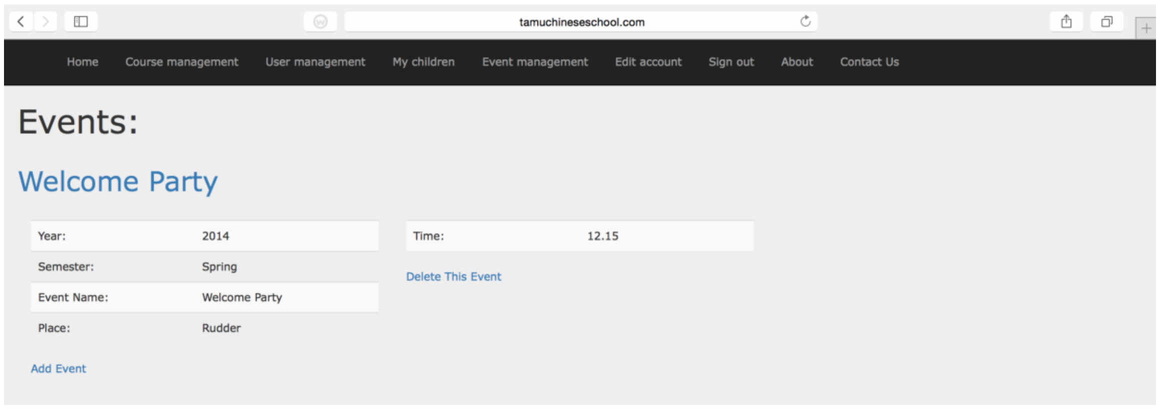


Listing all children

Name	Chinese Name	Gender	Course Taking	
Will SmothDelete	Will Smoth	Male	Chinese Drop	Math Drop
Lily SmithDelete	Lily Smith	Female	Software Engineering Drop	
Anna AdamsDelete	Anna Adams	Female	Math Drop	Chinese Drop
Peter ParkerDelete	Peter Parker	Male	Dancing Drop	

User Story: Add an event to the database


TAMU Chinese School at College Station
Listing all events
Events # 1
Year:
Time:
Semester:
Description:
Event Name:
Delete This Event
Place
Event # 2
.....
.....
.....
Add Event
TAMU Chinese School at College Station
Back To Event List
Events Title
Year: ....
Semester: ....
Event Name: ....
Place: ....
Time: ....
Description: ....
TAMU Chinese School at College Station
Register Event
Year
Event Name
Time
Place
Semester
Description
Submit



Events:

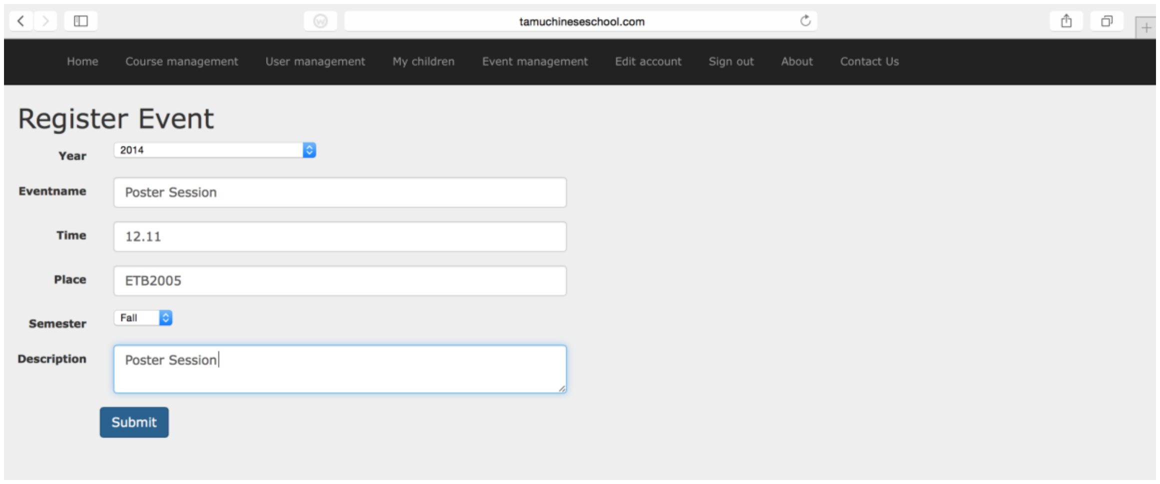
Welcome Party

Year:	2014
Semester:	Spring
Event Name:	Welcome Party
Place:	Rudder

Time:	12.15

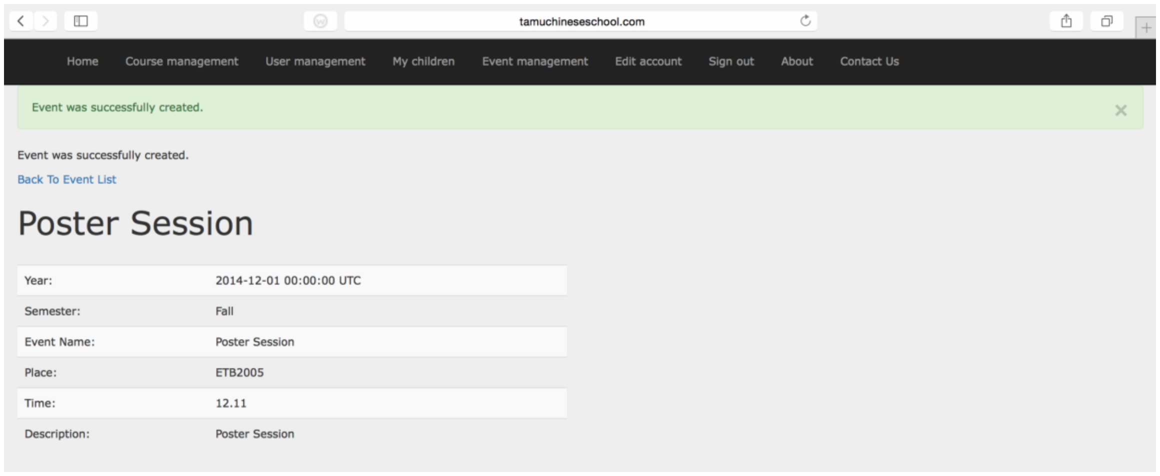
Delete This Event

Add Event



Register Event

Year	2014
Eventname	Poster Session
Time	12.11
Place	ETB2005
Semester	Fall
Description	Poster Session



Event was successfully created.

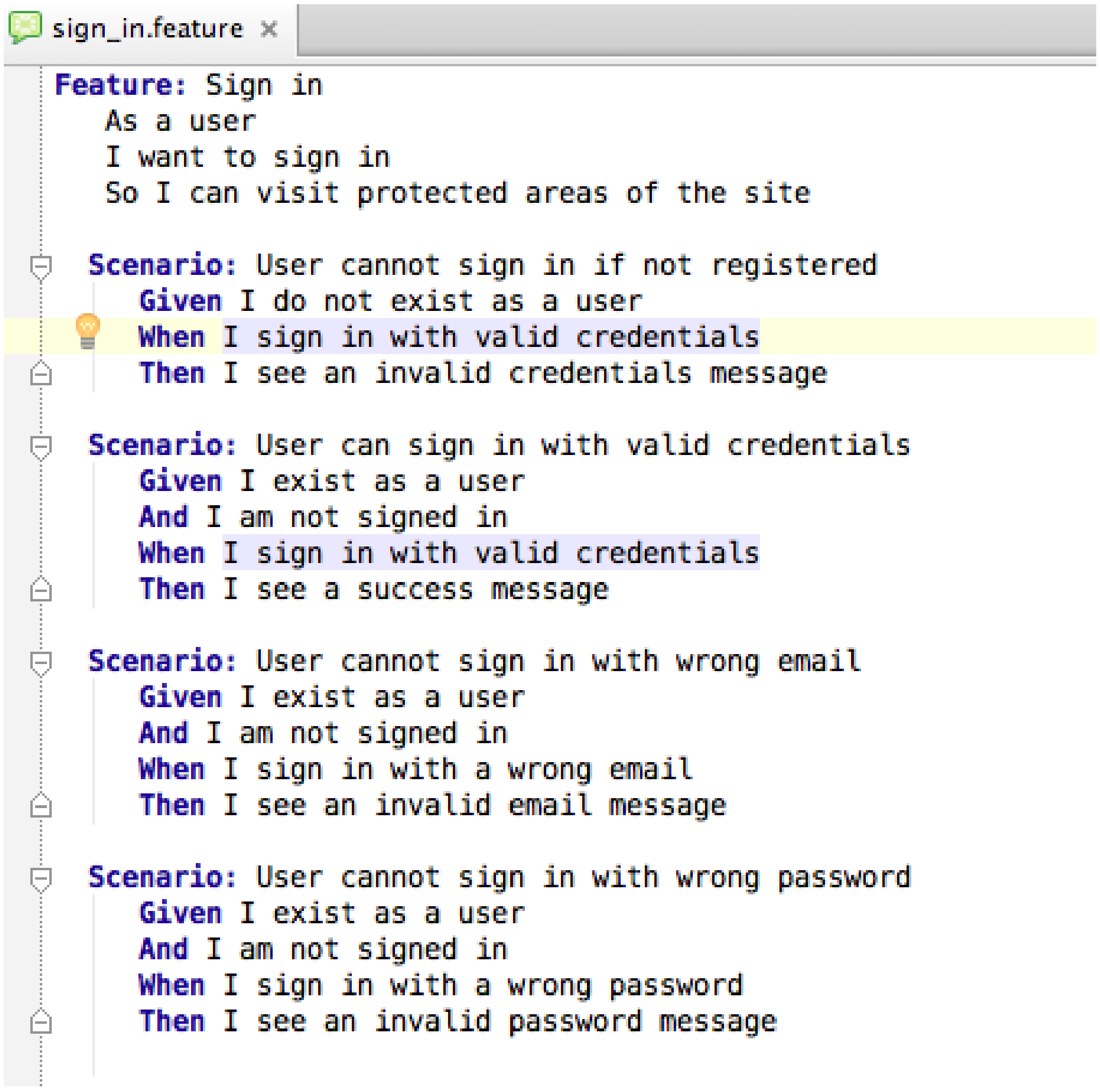
Event was successfully created.

Back To Event List

Poster Session

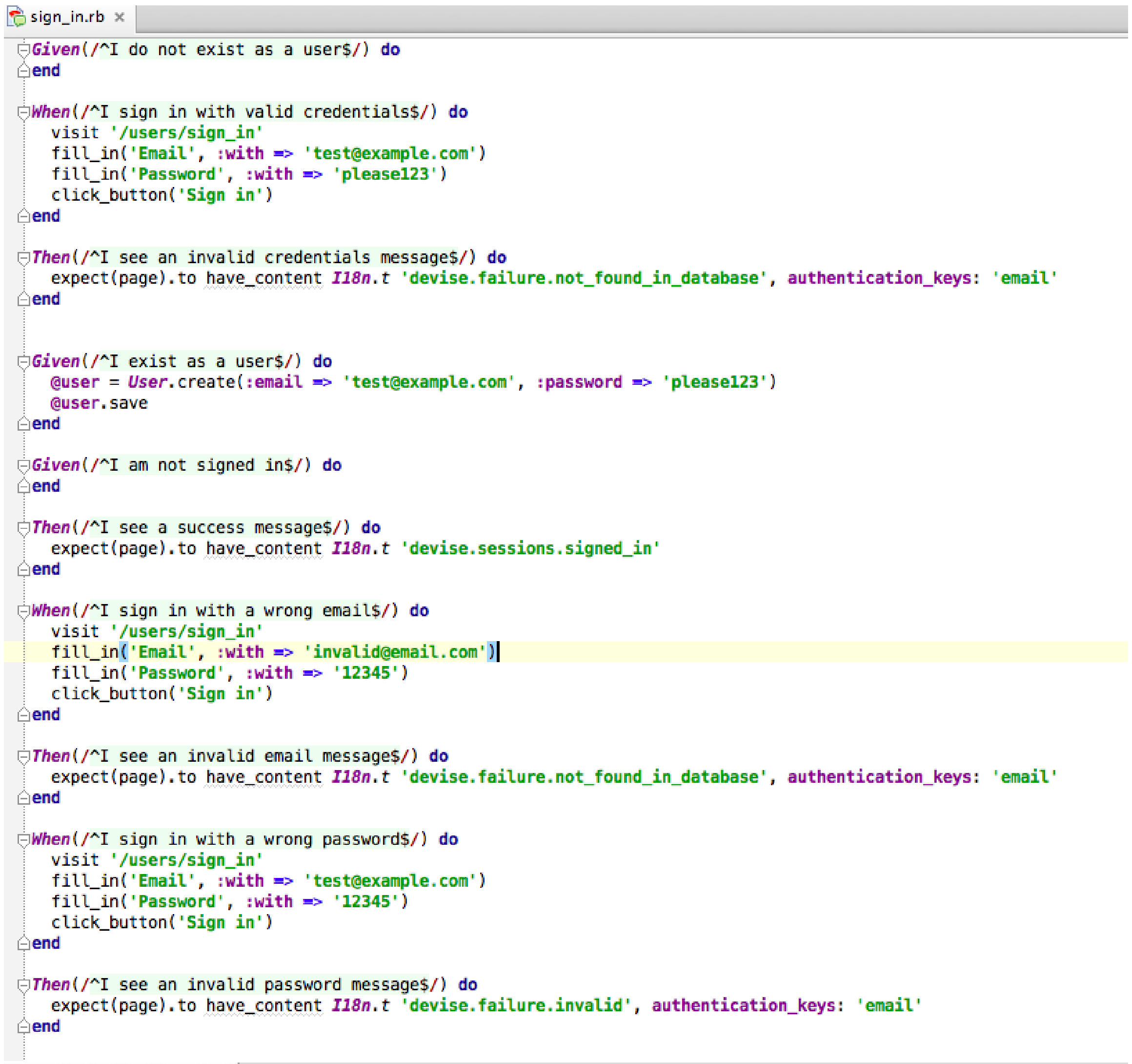
Year:	2014-12-01 00:00:00 UTC
Semester:	Fall
Event Name:	Poster Session
Place:	ETB2005
Time:	12.11
Description:	Poster Session

- BDD and TDD testing
  - Cucumber

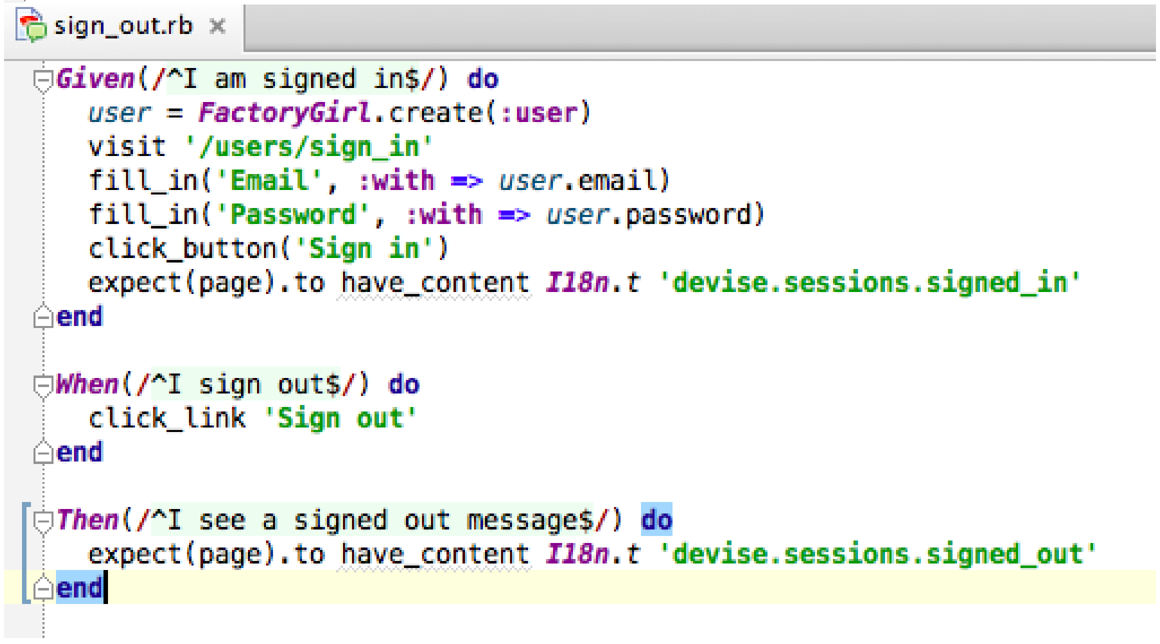


sign_in.feature

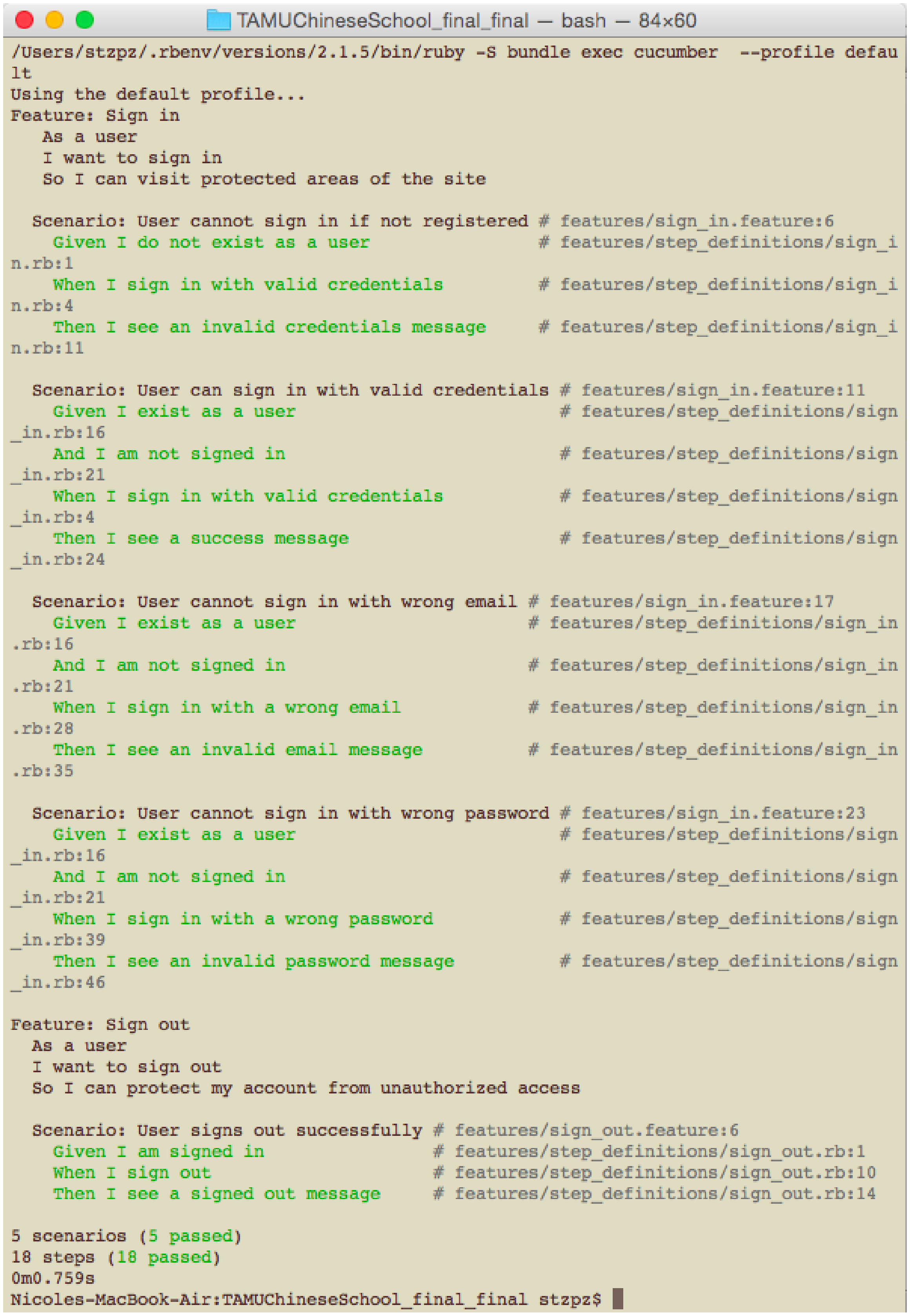
```
Feature: Sign in
   As a user
   I want to sign in
   So I can visit protected areas of the site

  Scenario: User cannot sign in if not registered
     Given I do not exist as a user
     When I sign in with valid credentials
     Then I see an invalid credentials message

  Scenario: User can sign in with valid credentials
     Given I exist as a user
     And I am not signed in
     When I sign in with valid credentials
     Then I see a success message

  Scenario: User cannot sign in with wrong email
     Given I exist as a user
     And I am not signed in
     When I sign in with a wrong email
     Then I see an invalid email message

  Scenario: User cannot sign in with wrong password
     Given I exist as a user
     And I am not signed in
     When I sign in with a wrong password
     Then I see an invalid password message
```



```ruby
Given(/^I do not exist as a user$/) do
end

When(/^I sign in with valid credentials$/) do
  visit '/users/sign_in'
  fill_in('Email', :with => 'test@example.com')
  fill_in('Password', :with => 'please123')
  click_button('Sign in')
end

Then(/^I see an invalid credentials message$/) do
  expect(page).to have_content I18n.t 'devise.failure.not_found_in_database', authentication_keys: 'email'
end


Given(/^I exist as a user$/) do
  @user = User.create(:email => 'test@example.com', :password => 'please123')
  @user.save
end

Given(/^I am not signed in$/) do
end

Then(/^I see a success message$/) do
  expect(page).to have_content I18n.t 'devise.sessions.signed_in'
end

When(/^I sign in with a wrong email$/) do
  visit '/users/sign_in'
  fill_in('Email', :with => 'invalid@email.com')
  fill_in('Password', :with => '12345')
  click_button('Sign in')
end

Then(/^I see an invalid email message$/) do
  expect(page).to have_content I18n.t 'devise.failure.not_found_in_database', authentication_keys: 'email'
end

When(/^I sign in with a wrong password$/) do
  visit '/users/sign_in'
  fill_in('Email', :with => 'test@example.com')
  fill_in('Password', :with => '12345')
  click_button('Sign in')
end

Then(/^I see an invalid password message$/) do
  expect(page).to have_content I18n.t 'devise.failure.invalid', authentication_keys: 'email'
end
```



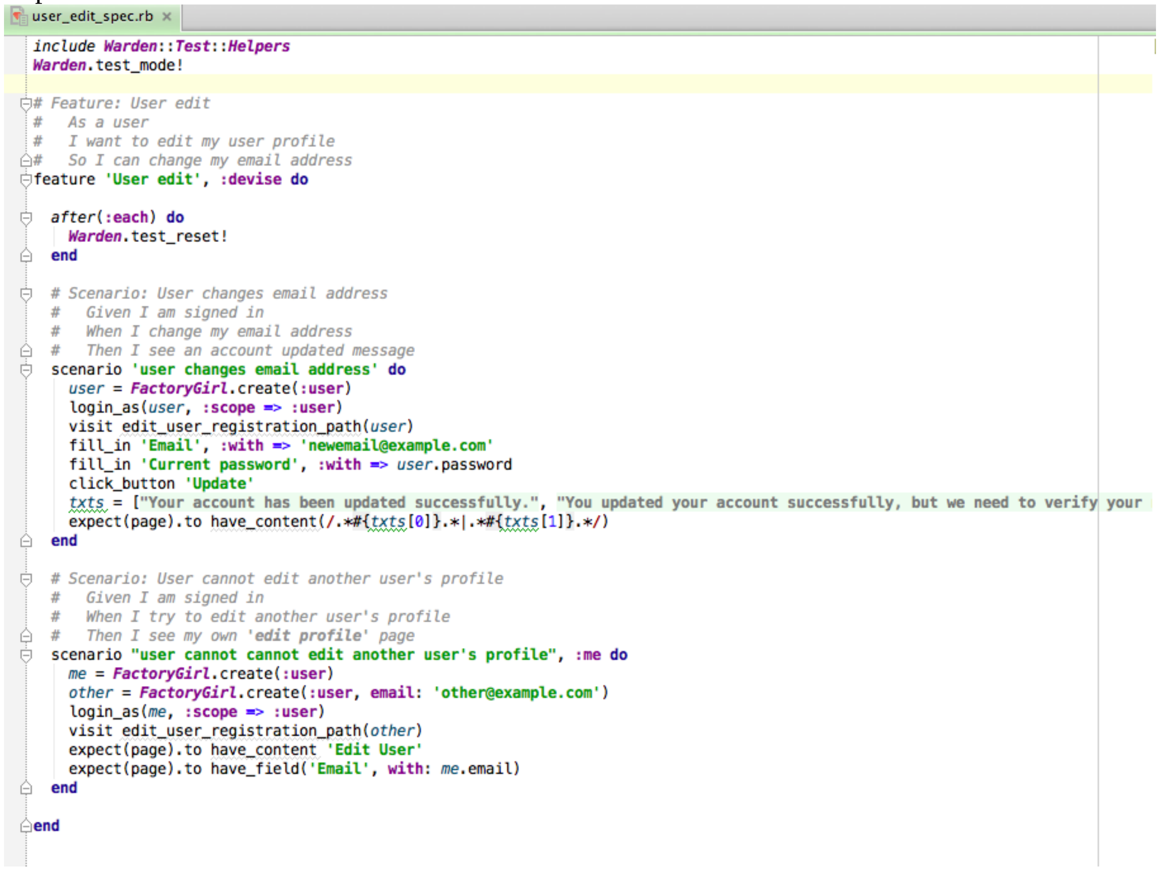
```ruby
Given(/^I am signed in$/) do
  user = FactoryGirl.create(:user)
  visit '/users/sign_in'
  fill_in('Email', :with => user.email)
  fill_in('Password', :with => user.password)
  click_button('Sign in')
  expect(page).to have_content I18n.t 'devise.sessions.signed_in'
end

When(/^I sign out$/) do
  click_link 'Sign out'
end

Then(/^I see a signed out message$/) do
  expect(page).to have_content I18n.t 'devise.sessions.signed_out'
end
```



TAMUChineseSchool_final_final — bash — 84×60

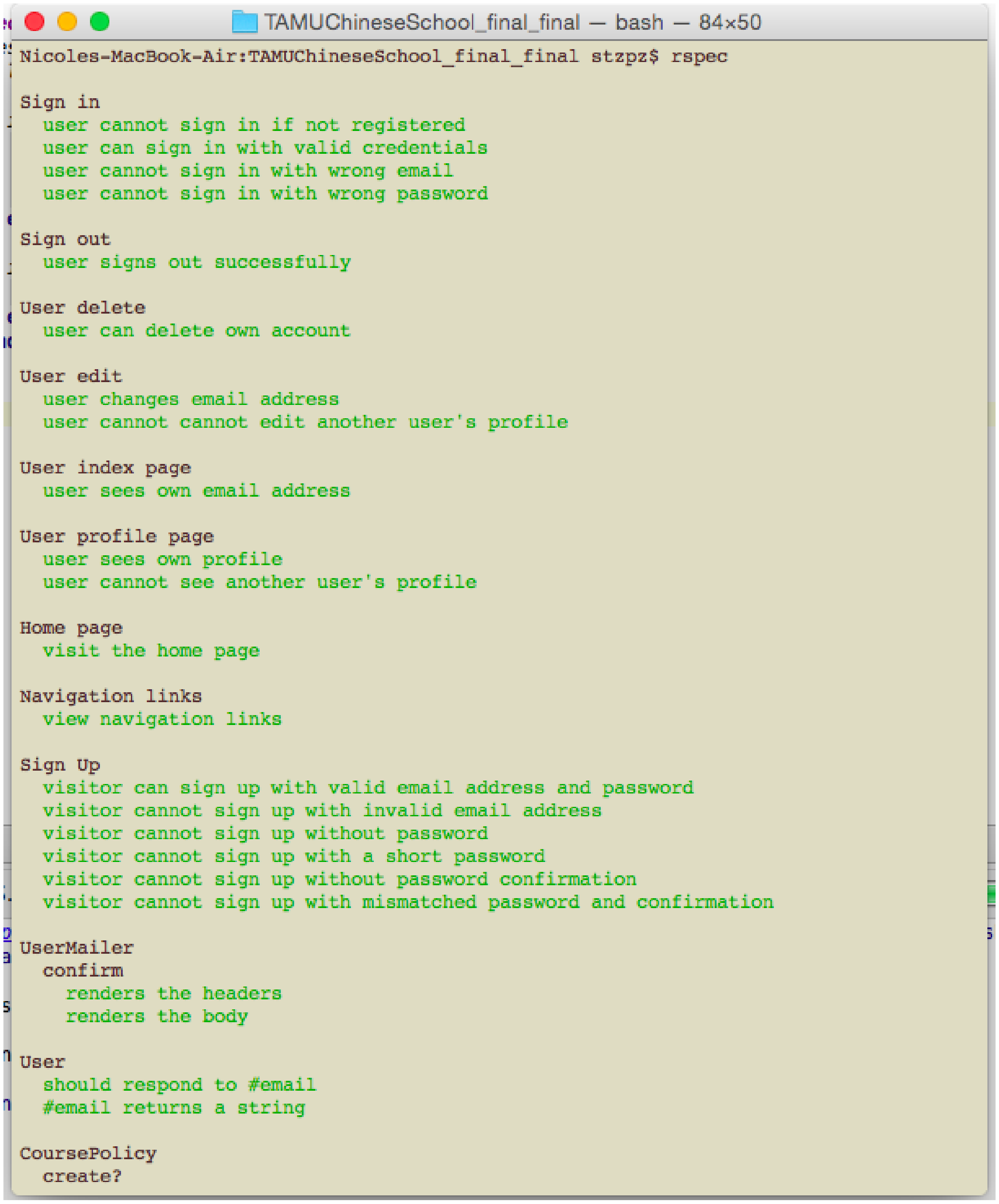
```
/Users/stzpz/.rbenv/versions/2.1.5/bin/ruby -S bundle exec cucumber  --profile defau
lt
Using the default profile...
Feature: Sign in
   As a user
   I want to sign in
   So I can visit protected areas of the site

  Scenario: User cannot sign in if not registered # features/sign_in.feature:6
    Given I do not exist as a user                # features/step_definitions/sign_i
n.rb:1
    When I sign in with valid credentials         # features/step_definitions/sign_i
n.rb:4
    Then I see an invalid credentials message     # features/step_definitions/sign_i
n.rb:11

  Scenario: User can sign in with valid credentials # features/sign_in.feature:11
    Given I exist as a user                         # features/step_definitions/sign
_in.rb:16
    And I am not signed in                          # features/step_definitions/sign
_in.rb:21
    When I sign in with valid credentials           # features/step_definitions/sign
_in.rb:4
    Then I see a success message                    # features/step_definitions/sign
_in.rb:24

  Scenario: User cannot sign in with wrong email # features/sign_in.feature:17
    Given I exist as a user                      # features/step_definitions/sign_in
.rb:16
    And I am not signed in                       # features/step_definitions/sign_in
.rb:21
    When I sign in with a wrong email            # features/step_definitions/sign_in
.rb:28
    Then I see an invalid email message          # features/step_definitions/sign_in
.rb:35

  Scenario: User cannot sign in with wrong password # features/sign_in.feature:23
    Given I exist as a user                         # features/step_definitions/sign
_in.rb:16
    And I am not signed in                          # features/step_definitions/sign
_in.rb:21
    When I sign in with a wrong password            # features/step_definitions/sign
_in.rb:39
    Then I see an invalid password message          # features/step_definitions/sign
_in.rb:46

Feature: Sign out
  As a user
  I want to sign out
  So I can protect my account from unauthorized access

  Scenario: User signs out successfully # features/sign_out.feature:6
    Given I am signed in                # features/step_definitions/sign_out.rb:1
    When I sign out                     # features/step_definitions/sign_out.rb:10
    Then I see a signed out message     # features/step_definitions/sign_out.rb:14

5 scenarios (5 passed)
18 steps (18 passed)
0m0.759s
Nicoles-MacBook-Air:TAMUChineseSchool_final_final stzpz$
```

Rspec:



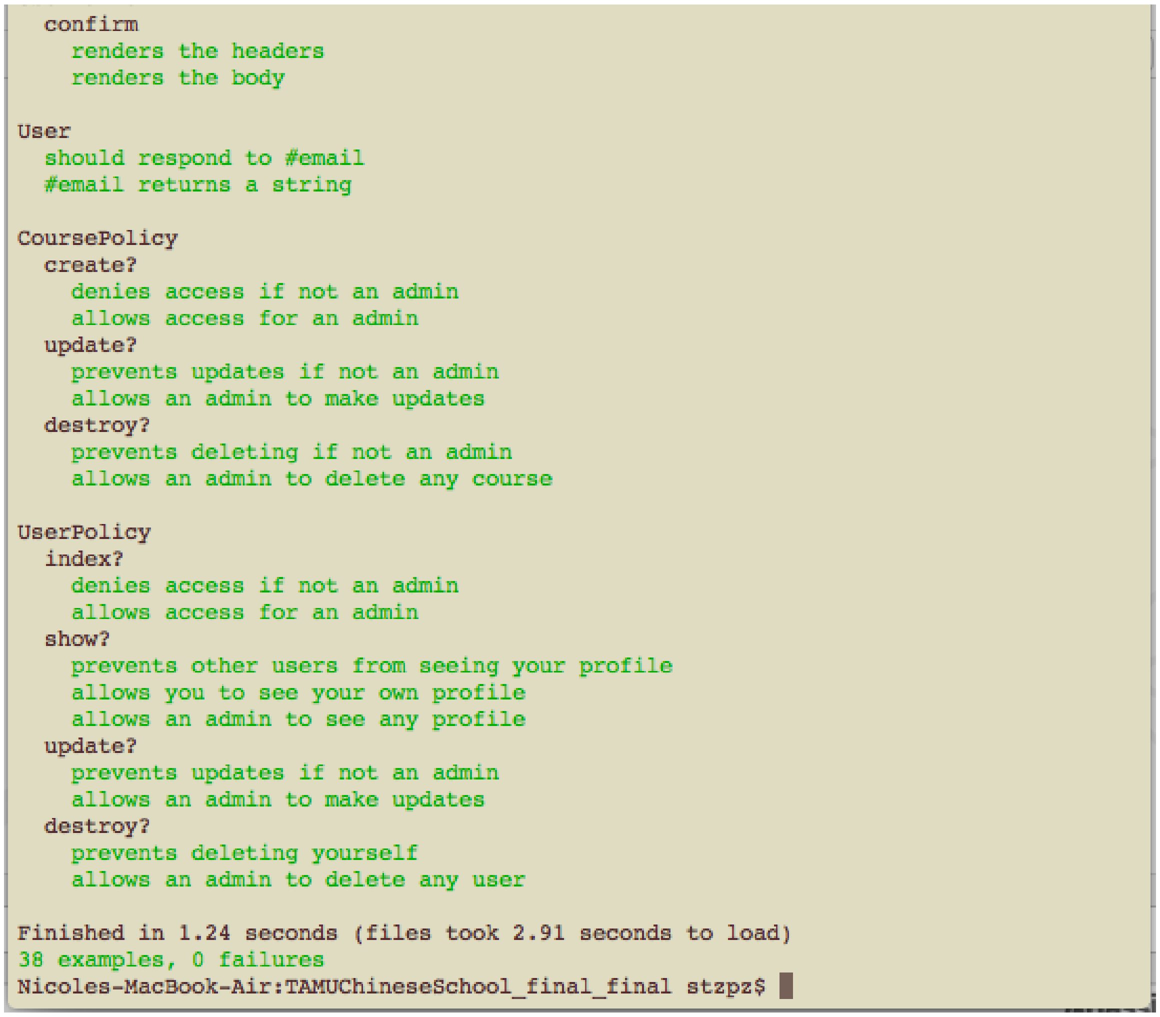
user_edit_spec.rb

```ruby
include Warden::Test::Helpers
Warden.test_mode!

# Feature: User edit
#   As a user
#   I want to edit my user profile
#   So I can change my email address
feature 'User edit', :devise do

  after(:each) do
    Warden.test_reset!
  end

  # Scenario: User changes email address
  #   Given I am signed in
  #   When I change my email address
  #   Then I see an account updated message
  scenario 'user changes email address' do
    user = FactoryGirl.create(:user)
    login_as(user, :scope => :user)
    visit edit_user_registration_path(user)
    fill_in 'Email', :with => 'newemail@example.com'
    fill_in 'Current password', :with => user.password
    click_button 'Update'
    txts = ["Your account has been updated successfully.", "You updated your account successfully, but we need to verify your
    expect(page).to have_content(/.*#{txts[0]}.*|.*#{txts[1]}.*/)
  end

  # Scenario: User cannot edit another user's profile
  #   Given I am signed in
  #   When I try to edit another user's profile
  #   Then I see my own 'edit profile' page
  scenario "user cannot cannot edit another user's profile", :me do
    me = FactoryGirl.create(:user)
    other = FactoryGirl.create(:user, email: 'other@example.com')
    login_as(me, :scope => :user)
    visit edit_user_registration_path(other)
    expect(page).to have_content 'Edit User'
    expect(page).to have_field('Email', with: me.email)
  end

end
```



TAMUChineseSchool_final_final — bash — 84×50

```
Nicoles-MacBook-Air:TAMUChineseSchool_final_final stzpz$ rspec

Sign in
  user cannot sign in if not registered
  user can sign in with valid credentials
  user cannot sign in with wrong email
  user cannot sign in with wrong password

Sign out
  user signs out successfully

User delete
  user can delete own account

User edit
  user changes email address
  user cannot cannot edit another user's profile

User index page
  user sees own email address

User profile page
  user sees own profile
  user cannot see another user's profile

Home page
  visit the home page

Navigation links
  view navigation links

Sign Up
  visitor can sign up with valid email address and password
  visitor cannot sign up with invalid email address
  visitor cannot sign up without password
  visitor cannot sign up with a short password
  visitor cannot sign up without password confirmation
  visitor cannot sign up with mismatched password and confirmation

UserMailer
  confirm
    renders the headers
    renders the body

User
  should respond to #email
  #email returns a string

CoursePolicy
  create?
```



```
  confirm
    renders the headers
    renders the body

User
  should respond to #email
  #email returns a string

CoursePolicy
  create?
    denies access if not an admin
    allows access for an admin
  update?
    prevents updates if not an admin
    allows an admin to make updates
  destroy?
    prevents deleting if not an admin
    allows an admin to delete any course

UserPolicy
  index?
    denies access if not an admin
    allows access for an admin
  show?
    prevents other users from seeing your profile
    allows you to see your own profile
    allows an admin to see any profile
  update?
    prevents updates if not an admin
    allows an admin to make updates
  destroy?
    prevents deleting yourself
    allows an admin to delete any user

Finished in 1.24 seconds (files took 2.91 seconds to load)
38 examples, 0 failures
Nicoles-MacBook-Air:TAMUChineseSchool_final_final stzpz$
```