Campus Web を作ろう班 報告書

立命館コンピュータクラブ Campus Web 班
城野 太河 *
青木 宏祐 †
迫 佑樹 ‡
音石 朋恵 §
小林 辰彰 ¶
鈴木 雅隆 ‖

2016 年 8 月 8 日

* 情報理工学部情報システム学科 2 回生
† 情報理工学部情報システム学科 3 回生
‡ 理工学部ロボティクス学科 2 回生
§ 情報理工学部メディア情報学科 2 回生
¶ 理工学部電子情報工学科 2 回生
‖ 情報理工学部工学科 1 回生

目次

1　はじめに

1.1　活動背景

立命館大学では，学生向けポータルサイトとして Campus Web と呼ばれるサービスを提供している．Campus Web を通して学生は講義情報の確認や履修登録，大学からのお知らせなどを受け取ることができるようになっている．そんな中，多くの学生から Campus Web が使いにくいといった声が上がっている．Campus Web は 3 万 5000 人を超える学生及び 3000 人以上の教職員が日々使用するサイトであり，更に利便性を高め，多くのユーザに快適に使ってもらえるよう改善する必要があると考えた．このような背景があり，Campus Web を新たに作るという活動を行うこととなった．

1.2　活動目的

多くの学生から使いにくいという声が上がっている現在の Campus Web について，ユーザが使いにくいと感じる部分について考察し，現在の Campus Web の必要な機能を移植しつつユーザビリティの高いサイトを作成することを目的として活動を行う．

2　現在の Campus Web の考察

現在の Campus Web に備えられている機能について確認し，その後に Campus Web のデザイン面での問題，システム面での問題を洗いだした後に，ユーザが使いにくいと感じる原因を考察した．

2.1　現在の Campue Web に備えられている機能

まずはじめに，現在の Campus Web にどのような機能があるのかを確認する．

1. 講義情報確認機能

休講情報や補講，教室の変更など，授業に関するお知らせを確認することができる．

2. 一般情報確認機能

学内で実施されているイベントや全学を対象に開催されている各種講演会及びシンポジウムのお知らせを確認することができる．

3. 履修登録及び確認機能

講義の受講登録手続きも Campus Web から行うことができる．受講登録の手続きや登録した時間割を確認することができる．

4. 試験情報確認機能

定期試験及びレポート試験の情報を確認することができる．持ち込み可能物件や試験時間割の確認ができ，学生への試験情報の開示はすべて Campus Web 上で行われる．

5. 個人情報紹介及び変更機能

本人の現住所，緊急連絡先を登録，変更することができる．また，クラブ活動や体育会活動などの所属団体に関する情報もここから登録することができる．

6. 学校関係のリンク集及びよく使うサイトのリンクの登録機能

オンラインシラバスや立命館大学が提供しているメールシステムへのリンク，学内情報環境使用方法の解説ページなど，学校生活を送る上で必要となるサイトへのリンクがまとめられている．また，よく使うサイトを自分で登録することもできるようになっている．

2.2 現 Campus Web のユーザ・インタフェイス

現在の Campus Web のユーザ・インタフェイス (以下 UI) についても確認しておく．
トップページは図 2.1 のようになっている．

図 2.1 Campus Web トップページ

トップページからお知らせ一覧が見れるようになっており，個人，授業連絡，履修，学生生活，進路，国際・語学，イベントとカテゴリ分けがされている．また，サイドバーに履修登録ページやテスト時間割紹介ページへのリンクが貼られている．

2.3 UI 面での問題点

1. 別ウィンドウで開く ... Campus Web は，アクセスする際に立命館大学の在学生向けのホームページからのリンクをたどる必要があり，その際 Campus Web のリンクをクリックするとに別ウィンドウが開いてしまう．
2. ブラウザバックの禁止 ... Campus Web ではブラウザの戻るボタンやキーボードの back space を用いて前のページに戻ることができない．戻るときはサイトに設けられている戻るボタンを押さなくてはならない．ブラウザの戻るボタンを使うと，図 2.2 のようなエラーメッセージが表示さ

れる．

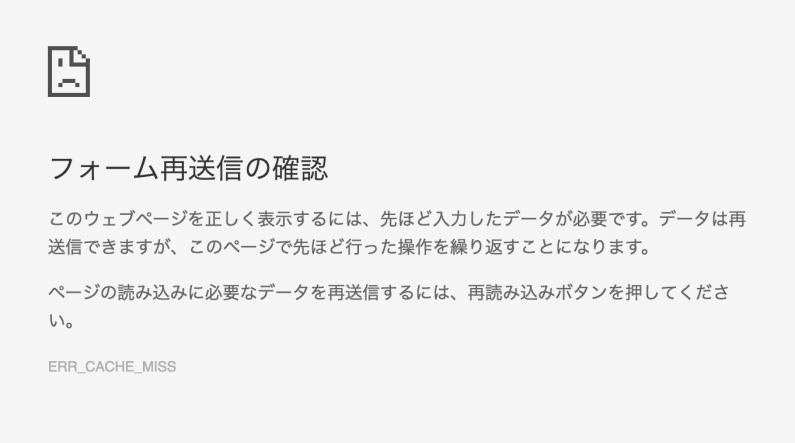

図 2.2　ブラウザの戻るボタンを使用した際のエラー

3. ブックマークの禁止 . . . Campus Web のトップページをブックマークしておくことは禁止されており，ブックマークから Campus Web へとアクセスした場合，図 2.3 のようなエラーが発生する．

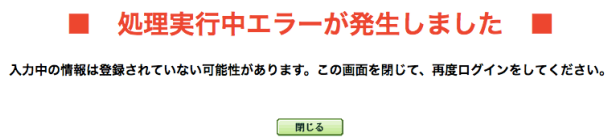

図 2.3　ブックマークからのアクセス時のエラー

このため，ブックマークをすることができず，ユーザにとってはわざわざ別ページからアクセスするという手間が生まれてしまう．

4. 右クリックが使えない . . . Campus Web 上では，右クリックが禁止されている．実際に右クリックをすると図 2.4 のようなエラーが表示される．

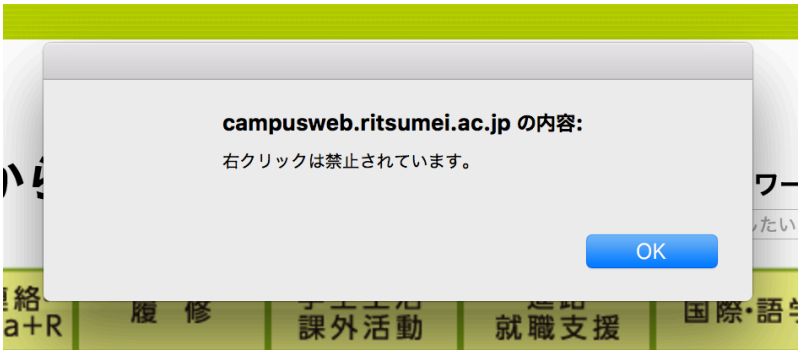

図 2.4　右クリック時のエラー

このため，ユーザは右クリックを用いた操作をすることができず，毎度右クリックを適用したい部分を選択し，ブラウザのメニューから操作しなければいけないという手間が発生する．

5. デザイン性の欠如 ... Campus Web はデザイン性に欠けている．見た目が野暮であるという点に加え，トップページでお知らせの一覧を見ることができないなどの問題点がある．また，トップページを開いた際に最初に出るウィンドウサイズがサイトの最小のサイズよりも小さく，右側が切れた状態で表示される．

このようなデザイン性の欠如はユーザにサービスの使いにくさを感じさせる要因の一つとなると考えられる．

6. 不要な「便利リンク」 ... Campus Web のトップページの下部に「便利リンク」というリンク集があるが，これらのリンクは立命館の「在学生の方へ」というページにあるものと同じものが多い．ログインしなくても見ることができるものをわざわざアクセスしにくい Campus Web にログインして見る可能性は低いと考えられる．よってこの「便利リンク」は不要である．
7. 新着通知一覧がない ... Campus Web では，最初からお知らせがカテゴライズされているため，時系列で見たり，全てのカテゴリーを一覧で見ることができない．
8. パンくずリストがない ... パンくずリストとは，上位の階層となる WEB ページを階層順にリストアップしてリンクを設置したリストのことである．ユーザーが今 WEB サイト内のどの位置にいるのかを視覚的に認識することができる．Campus Web にはパンくずリストが存在しないため，ユーザが自分の位置を認識しずらくなっている．
9. 通知が執拗にタブ分けされてる ... 前述したように，Campus Web ではお知らせが執拗にタブ分けされている．また，そのタブ分けはわかりにくいもので，必要な情報にたどり着きにくいものとなっている．
10. レスポンシブデザインになっていない ... Campus Web はレスポンシブデザインになっていないため，スマートフォンから見ると大変見づらい．スマートフォンが普及した現在では，そこから Campus Web を見る機会も多いと考えられるので，レスポンシブデザインに対応したサイトにする必要がある．

2.4 システム的な問題点

1. 文字コード

Campus Web には文字コードとして，Windows31j と呼ばれる文字コードが使用されている．Windows31j とは，マイクロソフト及び，MS-DOS の OEM ベンダが Shift_JIS を独自に拡張した文字コードである．これが文字化けの原因となっている．

2. 禁止文字チェッカー

Campus Web には禁止文字チェッカーと呼ばれる機能がある．文字コードとして Windows31j が使われているため，文字化けしてしまう文字が多くある．そのため，文章の中に使える文字が含まれているかどうかを確認する禁止文字チェッカーがあると考えられる．ユーザビリティの観点から考えると，このような禁止文字チェッカーをいちいち使用するのはよくない．

禁止文字の一部を図 2.5 に示す．

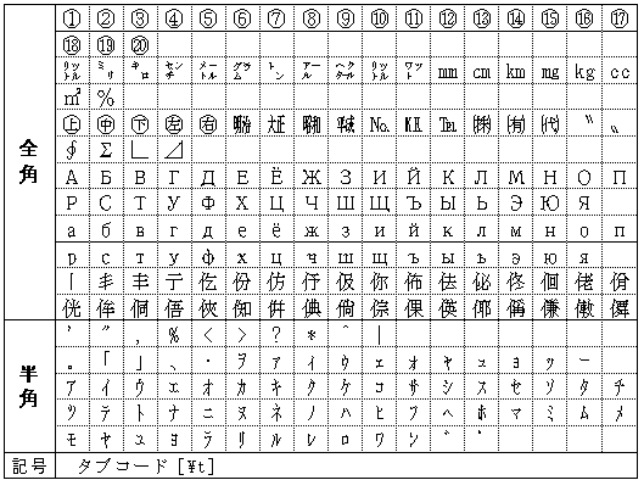

図 2.5　禁止文字の一例

3. .do がダウンロードされることがある

添付ファイルをダウンロードする際，拡張子が.do のファイルがダウンロードされてしまうことがある．本来は pdf ファイルがダウンロードされるべき場所で，図 2.6 のように，.do という拡張子のファイルがダウンロードされるため，ユーザは添付ファイルを確認できないことがある．

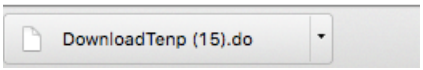

図 2.6　拡張子.do ファイル

.do とは，一般的には Java の Web アプリケーションフレームワークである”Struts” で使われる拡張子である．pdf ファイルをダウンロードする箇所において，.do がダウンロードされるのはバグである．

4. Cookie にユーザのパスワードが平文で保存されている

Cookie を確認してみると，ユーザのパスワードが平文で保存されていることがわかった．セッションハイジャック等のセキュリティの脆弱性になることが考えられるため，早急に改善が必要である．

2.5　ユーザビリティについての考察

以上の問題点について，ユーザが Campus Web を使いにくいと感じる原因を分析した．

1. システム状態の視認性欠如

ユーザのストレスを軽減するには，ユーザにシステムがなにをやっているかを知らせる．システムの視認性の例としてはネットからダウンロードした時のプログレスバーの表示などがある．Campus Web では，処理を行っている最中なのか処理中なのかどうかがわからず，処理を行っている最中に別の処理を行おうとすると，図 2.7 のように表示され，処理が止まってしまう．

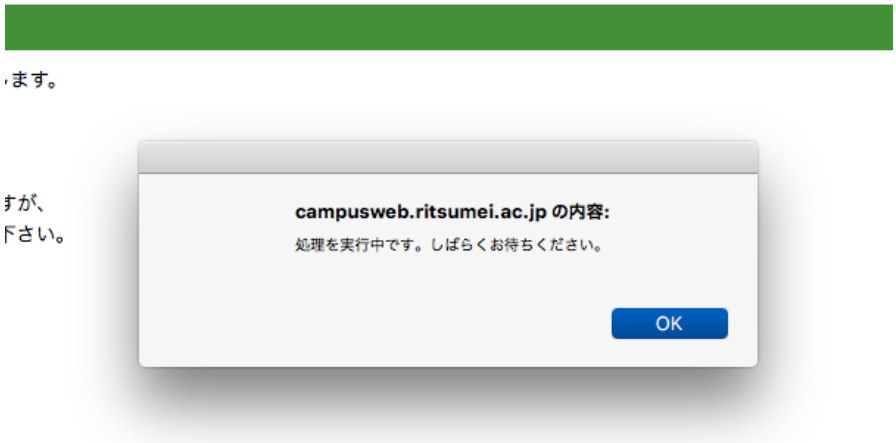

図 2.7　処理中のエラー

本来 Campus Web では時間のかかる処理は多くないため，プログレスバー等は必要ないが，処理の最中に別の処理を行った時にエラーが出るのはユーザにとって使いにくいと考えられる．

2. 習慣的行動との調和性欠如

Campus Web の使いにくい点として大きいのは，日常的に使用する一般のサイトでは行えることが禁止されているという点である．普段 Web ページを閲覧していて，ひとつ前のページに戻りたければブラウザに付いているバックボタンを使用し，新しいタブで開きたければ右クリックして “新しいタブで開く” ボタンを押す．よく使う Web ページはブックマークしておく．このように，ユーザが無意識のうちに使用する機能が Campus Web では禁止されている．これがユーザにとっては大きなストレスとなっていると考えられる．

3. デザイン性の欠如

デザイン性にかけるのもこのサイトが使いにくいところであると分析した．ブラウザバックを禁止しているにも関わらず，パンくずリストが上部に設置していなかったり，Top に戻るボタンがページ最下部に目立たないように書いてるなど，ユーザが使いにくいデザインとなっている．

4. 不要な機能の多さ

学校のポータルサイトにリンク機能は必要ない．Campus Web 自体がブックマーク不可能なのにもかかわらずわざわざ Campus Web を経由して他サイトへ行くことは考えられない．このように，いらない機能を Top ページに追加してしまっていることも使いにくい要因である．

以上 4 つが Campus Web が使いにくい原因であると私たちは考えた．この使いにくい要因を排除し，使いやすい Campus Web を設計していく．

3　設計・開発

3.1　開発手順

開発には，オブジェクト指向型プログラミング言語の Ruby 及び Web アプリケーションフレームワークである Ruby on Rails を使用する．

また，マークアップ言語には HTML5，スタイルシートには CSS3 を使用する．

3.2　フロントエンド設計

前述した UI の問題点を踏まえて，図 3.1 のようなデザインのトップページを作成した．

図 3.1　Campus Web トップページ

立命館のスクールカラーであるえんじ色を基調とし，UI に気を配ったデザインで作成した．新着のお知らせが最初に目にはいるようにし，メニューを右側に移動させた．その他の余分な情報は排除し，シンプルなものにした．

3.3　ユーザ管理

ここでは，新規登録，ログイン，ログアウトといった処理の実装について説明する．

3.3.1　データベース設計

Campus Web の利用者は，学生と教授の 2 種類にわけられる．学生と教授では，ユーザ名や Rainbow ID，パスワードなど共通した処理も多いため，User モデルを定義し，User モデルを継承した Student モデルと Professor モデルを定義することとした．

User モデルを図 3.2 に示す．

User モデルは，図 3.2 のように，User を一意に識別する id を持ち，ユーザ名を保存する name 属性，メールアドレスを保存する email 属性，パスワードを暗号化したものを保存する password_digest 属性を持つ．

この User モデルを継承した Student モデルを図 3.3 に示す．

Student モデルは，User モデルとひも付けを行うためのユーザ ID である user_id 属性，学籍番号を保存する student_number 属性，学年を保存する grade 属性，所属学部を保存する department 属性を持つ．

同じく User モデルを継承した Professor モデルを図 3.4 に示す．

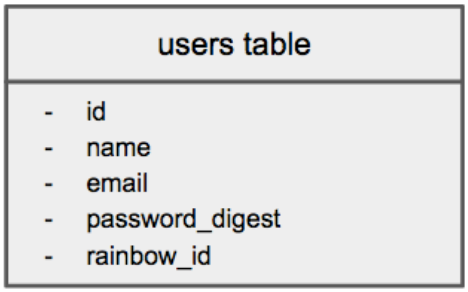

図 3.2　User モデル

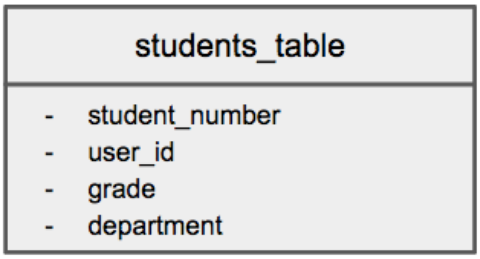

図 3.3　Student モデル

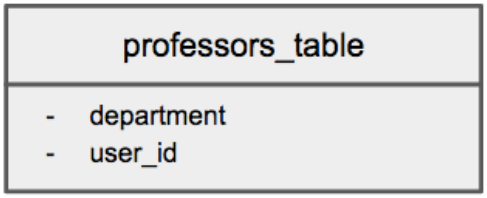

図 3.4　Professor モデル

Professor モデルは，User モデルとひも付けを行うためのユーザ ID である user_id 属性，所属学部を保存する department 属性を持つ．

3.3.2　新規登録処理

本来の Campus Web には新規登録機能はなく，入学時に発行される Rainbow ID を用いてログインを行う．今回はテストを兼ねてユーザを登録することとした．パスワードの暗号化には，bcrypt と呼ばれる gem を使った．

ユーザモデルを定義するファイルで，以下のように書くとパスワードを暗号して保存することができる．

```
class User < ActiveRecord::Base
  has_secure_password
end
```

この bcrypt を用いることで，以下の 3 つのことができるようになる．

- データベースにハッシュ化された password_digest を保存する．

- 新規登録時にパスワードと確認用パスワードの 2 つをユーザに入力してもらい，その 2 つが一致しているか検証する．
- パスワードが正しいときに、ユーザーを返す authenticate メソッドを追加する．

このように，bcrypt を使うことにより，ユーザ管理に必要な機能の多くを追加することができる．

3.3.3 ログイン・ログアウト処理

ログインする場合は，ログイン情報をセッションとして，サーバーとブラウザに保存する．

今回は，Rails で提供される session という特別な変数にユーザーの id を入れることで，ログイン機能を実現する．ログインが行われると，session にユーザ ID を格納し，ログアウトの際には session の中身を空にすることによって，どのユーザがログインしているのか，もしくはログインしてないのかを確認することができるようになっている．

3.4 講義情報確認機能設計

- index

講義の一覧を表示する．User は student モデルと professor モデルに分けられている．ログインしているのが student の場合，その student が受講登録している講義のみ表示し，professor の場合，その教授の担当講義のみ表示する

- show

選択された講義の詳細情報を表示する．講義名，曜日，時限などの詳細と編集，削除を行うリンクを表示する．

- new

新規講義を作成するためのページを表示する．講義作成ボタンが押されると入力されたデータを create に POST する．

- create

new から受け取ったデータをデータベースに新規レコードとして登録する．

- edit

選択した講義の編集フォームを表示する．編集完了ボタンが押されると入力したデータを update に POST する．

- update

edit で入力されたデータを受け取りデータベースの既存レコードを更新する．

- destroy

選択された講義のレコードをデータベースから削除する．

- assign

受講登録を行うためのページを表示する．全ての講義一覧を表示させ，選択した Lecture の id を register に POST する．

- register

ログインしているユーザーの student.id と assign から受け取った Lecture.id の 2 つを持ったレコードをデータベースに新規レコードとして登録する．

3.4.1 relation に関連する機能

生徒は受講登録を行った講義，教授は担当の講義をそれぞれ複数持っている．それら関連づけるために 2 つのデータベースを使用しており，中間テーブルの役割を担っている．データベースは student の id と Lecture の id，professor の id と Lecture の id の 2 つずつカラムを持っている．これにより生徒と講義，教授と講義の結びつけをしている．student がログイン中，student の id に結び付けられた Lecture の id を間接参照し，参照された Lecture を index に表示する．これにより，受講登録された講義のみを表示する．一方，professor がログイン中，同様に中間テーブルから professor の id に結び付けられた Lecture の id を間接参照することでその教授の担当講義のみを表示する．

3.4.2 ログインユーザに関する権限設定

- student

生徒は講義の作成，編集，削除は行わないため new，create，edit，update，destroy にはアクセス不可．

- professor

教授は受講登録は行わないため assign，register にはアクセス不可．

3.5 お知らせ機能設計

3.5.1 概要

ここでは CampusWeb におけるお知らせ機能の実装について説明する．

お知らせ機能を利用するユーザは学生，教授に分かれている．学生はお知らせ一覧と，お知らせ詳細画面を閲覧することができる．教授は閲覧機能に加え，新しいお知らせを追加，既存のお知らせを編集，削除することができる．

はじめに，お知らせ機能に必要な一覧表示機能，詳細表示機能，新規追加機能，編集機能，削除機能を実装した．

3.5.2 お知らせに必要な項目

お知らせ詳細画面は，お知らせのタイトル，本文，配信元，カテゴリ，添付ファイル，公開期間，お知らせ番号，状態を持つ．お知らせ詳細画面とお知らせ追加・編集画面に各々のフィールドを用意し，閲覧・登録できるようにした．

お知らせを追加・編集する際，タイトル，配信元，公開期間，お知らせ番号，状態はテキストフィールド (1 行) として入力でき，本文はテキストエリア (複数行) として入力できる．カテゴリはプルダウンで選択できる．添付ファイルはファイルを選択するボタンがあり，ボタンが押されるとファイルを選択する画面が表示され，開くとファイルが添付され，ボタンの隣にファイル名が表示される．

最後に公開するボタンを押すとお知らせ情報がデータベースに登録され，学生・教授が閲覧できるよう

になる．

3.5.3 バリデーション

各項目に入力できるフィールドには特定の条件がある．公開期間と状態は数値のみ入力可能である．カテゴリはプルダウンの選択になっており，通常の使用方法では必ずプルダウンのメニューからしか選択できないようになっているが，不正なスクリプトによってメニュー以外の値が入力されると空になり，カテゴリは空の状態では公開できないのでエラーになる．同様に，タイトル，本文，配信元，公開期間，お知らせ番号，状態のいずれかが空でアップロードされた場合もエラーになる．添付ファイルに関してはすべてのお知らせに必要とは限らないので，空でも公開できるようにした．

エラーが発生すると再び追加・編集画面が表示され，エラーの内容が表示される．

3.5.4 学生と教授の関連付け

ここまでの説明の実装では，教授が公開したすべてのお知らせが全学生・教授に閲覧できるようになっている．学生にとっては，履修していない授業のお知らせも表示される状態であり，教授にとっては，担当していない授業のお知らせも表示される状態である．

学生が履修している授業のみ，また教授が担当している授業のみを表示させるために，ログインしたユーザに関係のあるお知らせのみを表示するように関連付けを行った．

学生とお知らせの関係，および教授とお知らせの関係は多対多である．学生は複数のお知らせを受け取り，お知らせは複数の学生に送信される．また，教授は複数のお知らせを送信し，一つのお知らせが複数の教授によって書かれることもあることを想定している．

多対多の関係をモデルで表現するためには中間モデルが必要となる．学生とお知らせの関係をつなぐための中間モデルとして student_announcement を用意し，教授とお知らせの関係をつなぐための中間モデルとして professor_announcement を用意した．

学生は student_announcement を通してお知らせを参照し，教授は professor_announcement を通してお知らせを参照する．お知らせは student_announcement を通して学生を参照し，professor_announcement を通して教授を参照する．student_announcement は学生とお知らせに所属し，professor_announcement は教授とお知らせに所属している．以上により，学生・教授がお知らせと多対多の関係付けを持つことができる．

3.5.5 権限

ログインユーザが学生か教授かによって権限を分ける機能を実装した．ユーザが教授であればお知らせ一覧画面に新しいお知らせを追加するフォームへのリンクと，各お知らせの編集・削除のリンクを表示し，学生であればそれらのリンクが表示されないように実装した．

ただし，リンクからページ遷移せずに追加・編集ページに直接アクセスした場合には学生でもアクセス可能となってしまうため，追加・編集ページにアクセスしようとした場合にはユーザが学生か教授を判定し，学生であればエラーを表示し，ページアクセスできないようにした．

3.5.6 カテゴリ別表示

お知らせ一覧ページでは，その学生・教授に関係するお知らせが表示されるが，カテゴリごとにお知らせを表示する機能を実装した．お知らせ一覧ページにカテゴリのプルダウンがあり，カテゴリを選択するとそのカテゴリと一致するお知らせのみが一覧で表示される．たとえばプルダウンから「個人」のカテゴリを選択すると，カテゴリで設定した値が「個人」に該当するお知らせを検索することができる．

3.5.7　その他

添付されたファイルはお知らせ詳細からダウンロードできるように実装した．その他，データベース内ではカテゴリは英語表記で保存されているが，実際に表示するときには日本語で分かりやすく表示されるような機能を追加したり，ログインしていないときにはログインページに移動するようにするなどの細かい調整を行った．

3.6　掲示板機能設計

本節では，ユーザー同士の情報交換，共有を目的として実装した掲示板機能におけるデータ処理，擬似的なログイン認証，投稿内容の新規作成，編集・削除について説明する．

3.6.1　データベース設計

本機能では，文章の投稿に必要となる基本的な要素を Miropost モデル内で定義し，投稿者をログインユーザー名から取得だできるよう，User モデルと Micropost モデルを中継する Usermicropost モデルを定義した．

Micropost_table
id
title
message
messanger

表 3.1　Micropost モデル

Usermicropost_table
user_id
micropost_id

表 3.2　Usermicropost モデル

Usermicropost モデルは User クラスの持つ ID と Micropost クラスのもつ ID を一意的につなげるため,user と micropost の両方の id 属性を継承する. これにより，新規投稿時に投稿者名にログインユーザーの名前を自動的に使用させることができる．

3.6.2　ユーザーの投稿一覧表示機能

投稿自体の ID 以外を抜き出して表示させる操作は RESTful な設計で実装しきれなかった. この機能を実装するため, 以下の通り user_id の値をパラメータとして使用するルーティングを別に設計した.

```
Rails.application.routes.draw do

    resources :microposts

    get "microposts/user/:user_id, "to: 'microposts#user'
```

3.6.3 ログインアカウント名を用いた擬似認証

投稿の編集及び削除を投稿者本人のみが行えるよう，投稿時に User クラスから取得されている投稿者名と投稿を閲覧しているユーザー名から閲覧者と投稿者が同一であるかを判定する．

```
<% if @post.messanger == current_user.name%>
  <%= link_to 'edit', "/microposts/#{@post.id}/edit"%>
  <%= link_to 'Delete' ,@post, method: :delete%>
<% end %>
```

他のユーザーが同姓同名である場合を除けばこの判定で十分であるが，あくまで擬似的なものである．新規作成時に投稿したユーザーの ID を Micropost モデルに取得し，上の判定と同様にログインユーザーの ID と照合させるのが現実的であると考えられる．

4 考察・まとめ

4.1 現 Campus Web との比較

1. システム状態の視認性欠如フラッシュメッセージを使用して，処理の完了などを分かりやすく，かつスマートに通知する．
2. 習慣的行動との調和性欠如 CampusWeb の主要な機能を保持しつつ，現代的な Web アプリケーションとして再構成した．これにより，
   - 新しいタブ，新しいウィンドウ
   - ブックマーク機能
   - バックスペースによるページバック

   といったブラウザの機能を正常に利用できるようになった．
   また，CampusWeb の仕様として存在していた
   - 文字コードが Windows31j
   - やり取りされるファイルの拡張子がフレームワークで中間利用される「.do」

   といった点も，それぞれ UTF-8 を使用，すべてのファイルはそのままの拡張子で運用することで対応している．
3. デザイン性の欠如 HTML3 全盛時代のような古めかしいデザインスタイルは一掃し，3.2 節で述べた通りモダンなデザインを採用した．トップページ左上に無意味に陣取っていたキャン子とキャン坊[*1]はダイナミックリストラし，代わりに弊学ロゴをあしらっている．この後釜としては本会非公式マスコットキャラである「オロロちゃん」を想定している．また，パン屑リストを全てのページに実装し，ページ間の移動がよりスムーズに行えるように計らった．
4. 不要な機能の多さ旧 CampusWeb に搭載されていた「便利リンク」「禁止文字チェッカー」といった不要な機能は削除した．これらは学生利用のポータルサイトには不要なもの，サイトの非効率な構成により生じたものであった為，削減による影響は生じていない．

4.2 結論

前項で述べた通り，洗いだした問題点に対しては本プロジェクトで対策を講じた．ユーザビリティの改善を定量的に評価することは難しいが，悪い点をなくすことによって「使いにくさ」は削減できたものと

[*1] 公式名称である

考える．

4.3 展望

本プロジェクトにおいて作成した Web サービスには，いくつか課題が残っている．今後の展望として，それらを解決することが挙げられる．

- 高トラフィックへの対応

テストは行っていないが，数万人規模のユーザーによる高いトラフィックには耐えられないことが考えられる．Apache や Nginx 等を使用して多量のリクエストを捌く，キャッシュを利用してサーバーへのアクセス自体を削減するといった対応を考えている．

- 既存データベースとの対応

弊学で現在運用されているデータベースの構成が分からないため，対応は事実上不可能であると考えられる．この問題は実際の導入時に解決する必要がある，