**スクリプト言語プログラミングー最終レポート**

**マイカレンダー**

氏名：グエン・タイン・コン

学年：３年

学籍番号：71070072

　ログイン名：t10507tn

1. プロジェクトの定義、動機と目標
2. プロジェクトの定義

・個人の時間管理システムである

・具体的には以下の二つに分担する

—自分の時間割を決め、ウェブ上で管理する

—自分の目標を達成するために、自分の時間割はどうすればよいかもシステムが管理する

1. 動機

　これまでは、スケジュールを厳しく管理しなくても問題なく生活ができた。しかし、今学期から研究会に入ったことで非常に忙しくなり、時間の管理が難しくなった場面が何度もあった。そこで、自分の時間割をコントロールできるように、時間割を管理するソフトウェア（マイカレンダーのシステム）を作りたいと思った。

1. 目標

以下のシステムの機能を実現したい

　　・ログインの機能

　　　個人情報を管理するために、ログイン機能が必要である。ユーザがログインした後、自分のページ（時間割）を直接見ることができる。

　　・時間を管理する機能

　　　できればグーグルカレンダーのように作りたい。具体的にはカレンダーを表示したり、時間割に自分のタスクを追加、編集、閲覧したりできるようにしたい。　　　　　　タスクには、メモや始まりの時間と終わりの時間などの機能が必要である。タスクが始まる１時前に、システムがユーザに知らせるメールを送る。

　　・目標を目指す機能

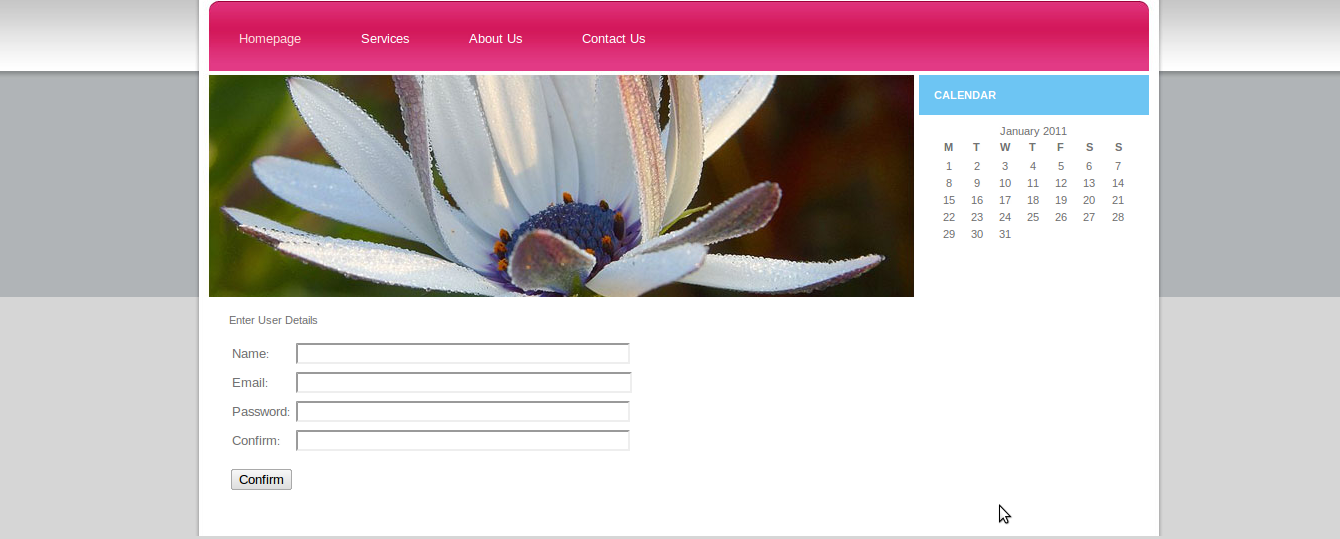
　　　ある時間帯ごとにしたいことを決め、それぞれの設定時間以内にタスクを終了させることを目指すことを目標とする。そして、その目標を達成するためにはどんなことをしなければならないかを考える。即ち、目標の中の目標が作成できるようにする。あるタスクを行っているときに、ユーザがそのタスクの目標の表示を見ると、その目標が達成可能かどうか、そのためにはどうすればいいか、不可能ならばその理由は何か、をシステムが自動的に答えるようにする。

1. 作成したアプリケーションの使い方の説明

ア）レジストレーション

初めてこのウェブを使用する場合、レジストレーションというステップが必要。自分のメールや名前といった簡単な個人情報を入力し、それから自分の好きなパスワードを入力すればレジストレーションのステップが終了。

下の画像はレジストレーションの画像を示す。

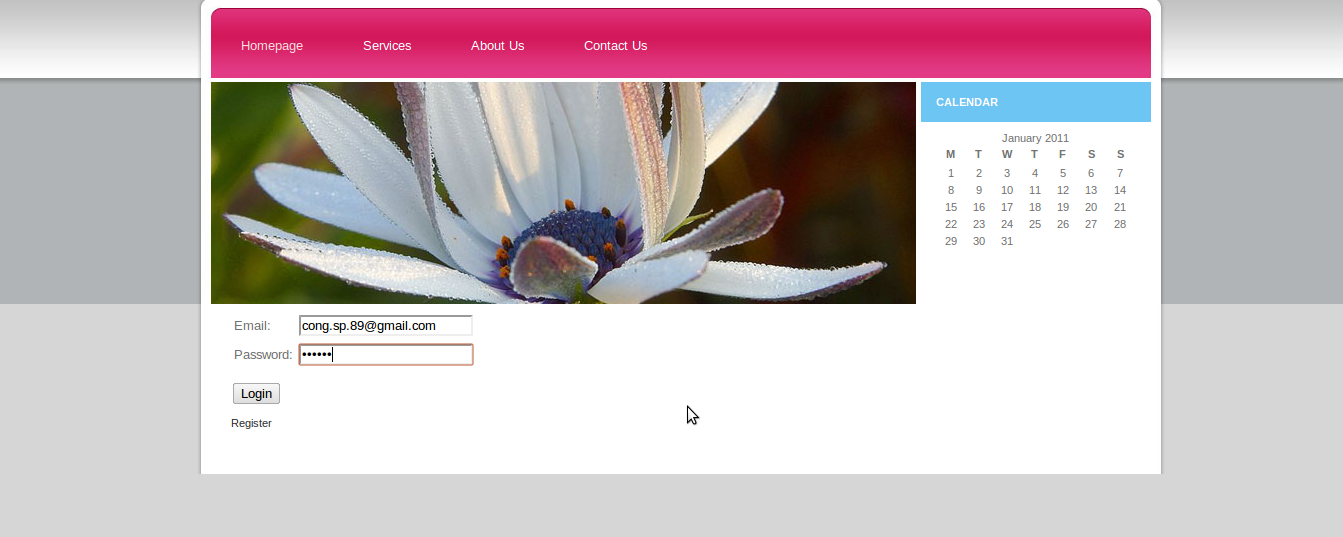


イ）ログイン

レジストレーションのステップが終わると、登録したメールアドレスにレジストレーション終了のメールが送られる。そこで、カレンダーウェブにログインすることができるようになる。

以下の画像はログインの画面である。

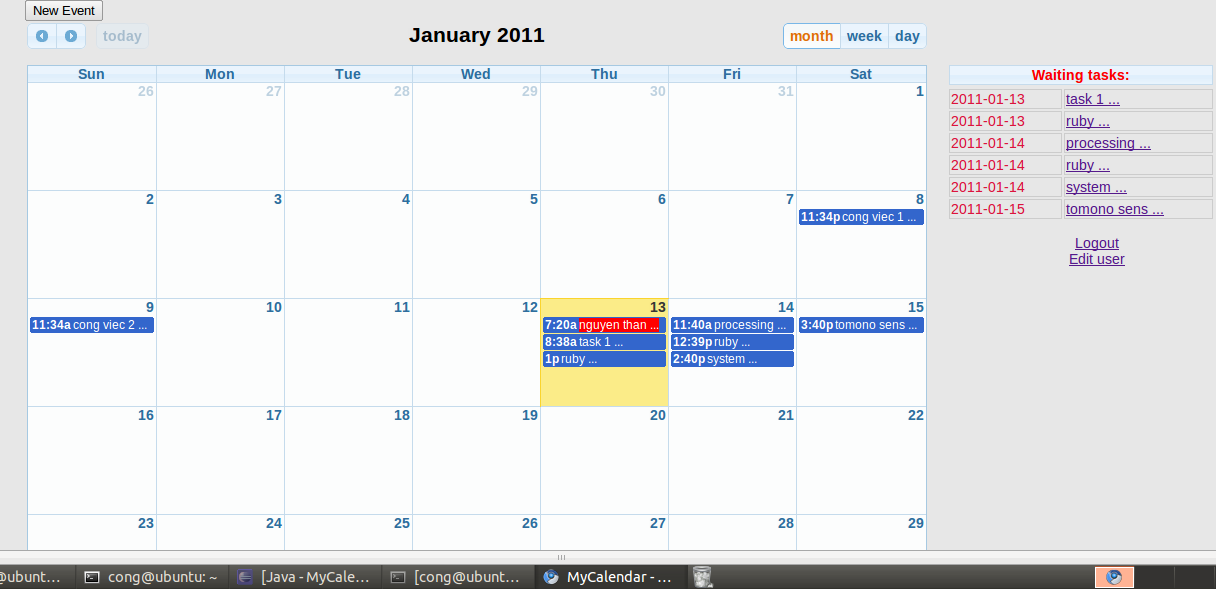
メールとパスワードの簡単な情報を入力すると、ログインできる。



ウ）タスク

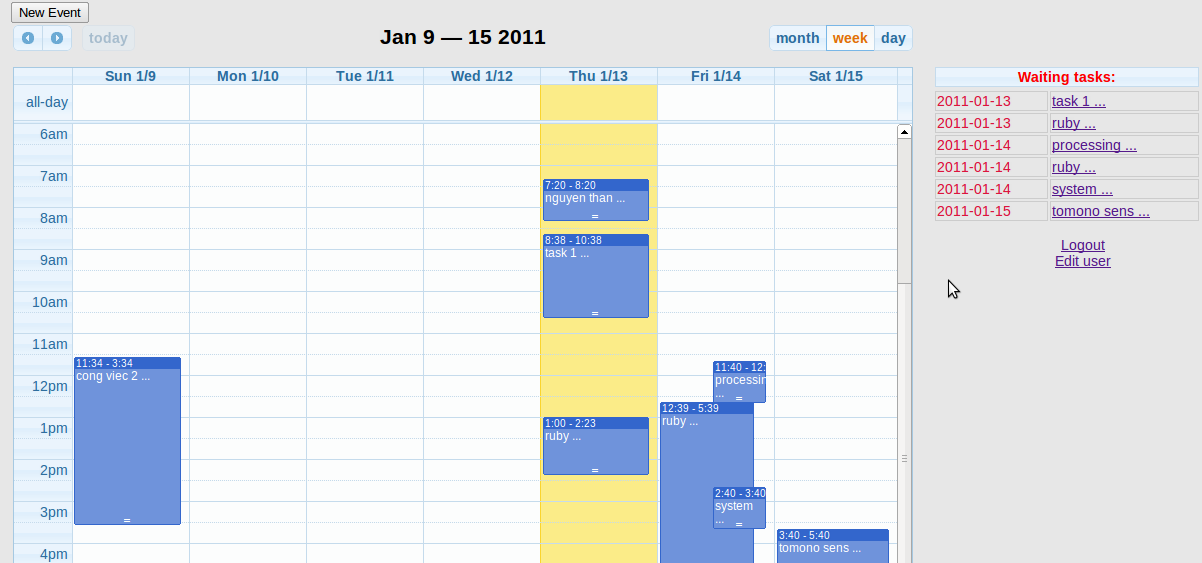
* ログインした後、自分のカレンダーとタスクの画像が表示される。カレンダーは月別のものである。
* 左側にあるNew eventのボタンを押すと、新しいタスクの情報をインプットする画面が出てくる。
* また、New eventボタンの下にある矢印をクリックすると、前の月のカレンダーや次の月のカレンダーが表示される。
* 中央にあるmonth, week, dayのタブをクリックすると、月のカレンダー、週のカレンダー、日のカレンダーがそれぞれ出てくる。
* 右側にあるwaiting taskのテーブルは、まだ終わっていないタスクを示す。
* 右下にあるlogoutのリンクをクリックすると、ウェブからログアウトできる。
* 右下にあるuser editのリンクをクリックすると、自分のパスワードを変更する画面が出てくる。

（月別のカレンダー）

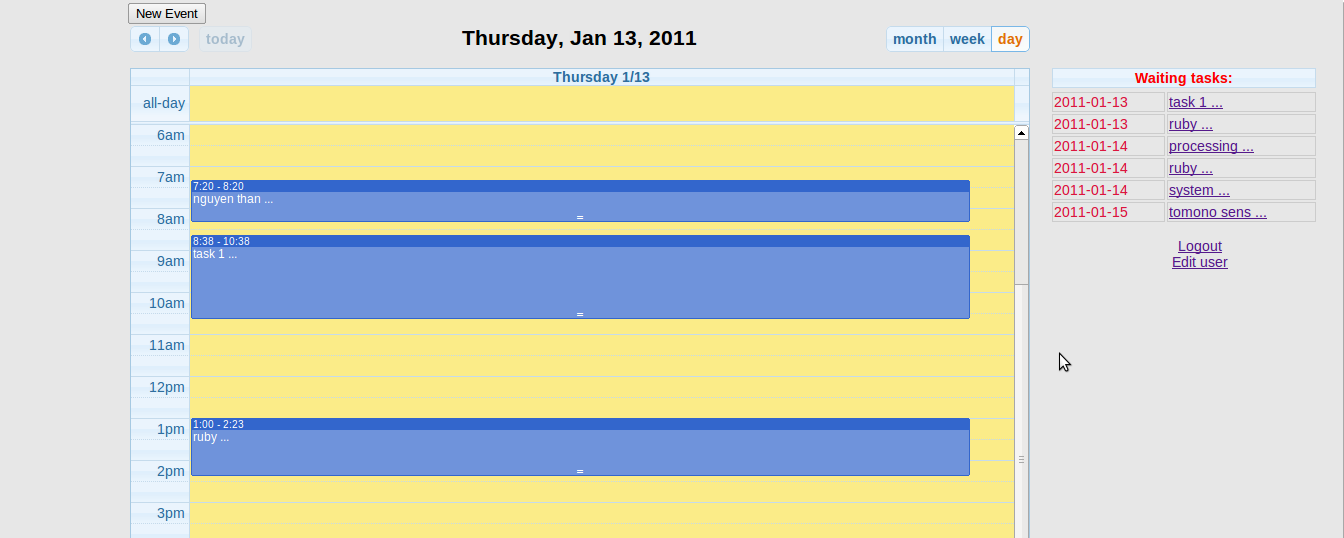


エ）タスクを入力すると、それがカレンダーに表示される。

（週別のカレンダー）



（日別のカレンダー）

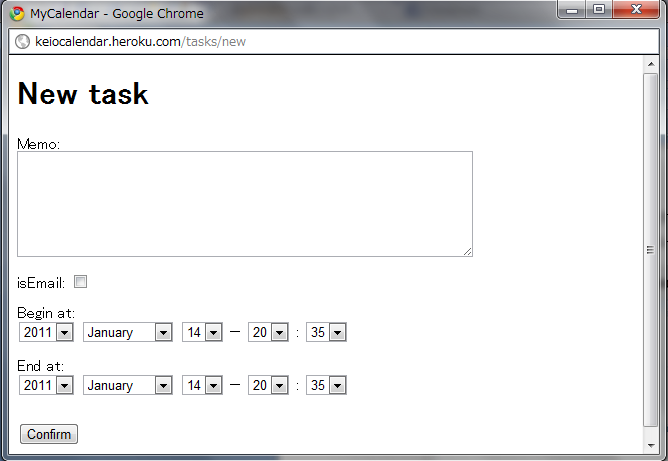


オ）新しいタスク

新しいタスクがあれば、カレンダーに入力できる機能である。以下の画面はタスクのフォームを示す。

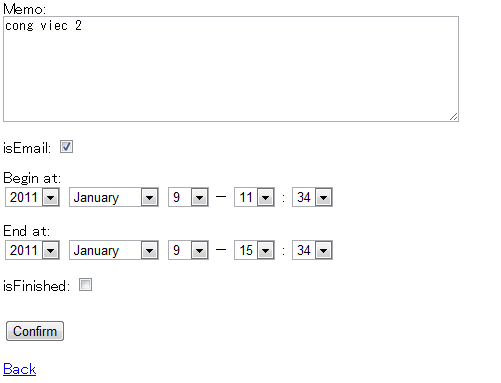
・Memoにはタスクの必要な情報を入力。

* Is Emailにはタスクが始まる一時間前にお知らせのメールを受け取るかどうかのチェックをする。
* Begin atとEnd atは、そのタスクの始まりと終わりの時間を指定。

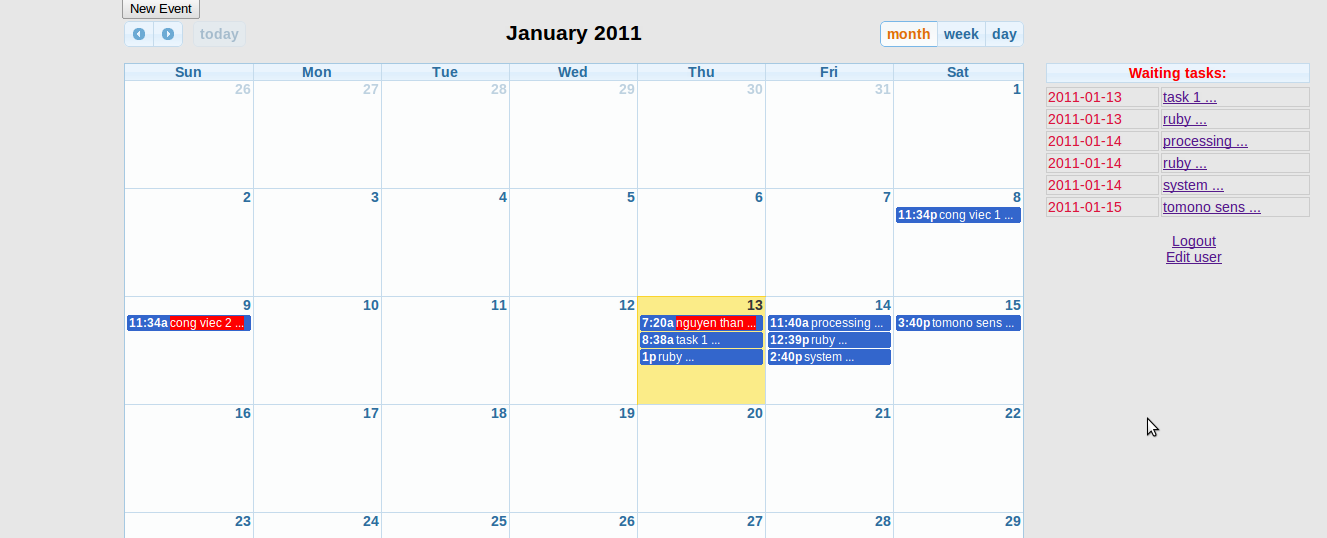


カ）タスクが終わった情報の確認

　あるタスクが終わったとき、カレンダーでその情報を確認することができるようにすることで管理しやすくする。カレンダーにあるその終了したタスクをクリックすると、タスクの情報が表示される。Is Finishedにチェックをいれると、そのタスクが終了したことになる。



＊終わったタスクはカレンダーに赤く塗られる。



カレンダーの情報が見終わったら、logoutをクリックして閉じる。

３．アプリケーションの内部構造の説明

ア）Model

・目標モデルにはユーザーによって様々な目標を持つことができる。

class Goal < ActiveRecord::Base

belongs\_to :user

end

・タスクモデルにはユーザーによって様々なタスクを持つことができる。

class Task < ActiveRecord::Base

belongs\_to :user

end

・ユーザーモデルにはユーザーによって様々なタスクを持つことができる。

class User < ActiveRecord::Base

has\_many :tasks

end

イ）View

タスクビュー

\_form.html.erb

\_formNew.html.erb

Delete.html.reb

Edit.html.erb

Index.html.erb

New.html.erb

Show.html.reb

ユーザーメール

Resgistration\_confirmation.text.erb

Task\_confirmation.text.erb

ユーザー

\_form.html.erb

afterRegister.html.erb

edit.html.erb

login.html.erb

new.html.erb

show.html.erb

ウ）Controller

Application\_controller.rb

Tasks\_controller.rb

Users\_controller.rb

エ）カレンダーを表示するビュー

Publicにjavascriptsとstylesheetsのフォルダがあり、

Javascriptsには

Fullcalendar.js

Fullcalendar.min.js

Jquery-1.4.4.min.js

Stylesheetsには

Fullcalendar.css

Theme.css

というファイルがある。

ビューにこれらファイルを呼び込むことで、カレンダーを表示。

オ）メールを送る機能

Config/initializersにsetup\_mail.rbファイルを置いた。

Libにdevelopment\_mail\_interceptor.rbファイルを置いた。

　この設定を行うことで、メールを送ることができるようになる。

　また、指定時間にメールを送る機能があるので、２分置きにデータベースをチェックし送らなければならないタスクがあればメールを送る。

　この機能を実行できるのはunixにあるcrontabというコマンドである。

　Crontabをする前に、App/models/task\_discover.rbというファイルを準備する。

　この準備ができたら、crontabの設定をする

　Crontab –e

\*/02 \* \* \* \* \* rails runner app/models/task\_discover.rb

これら設定が終わると、お知らせメールが自動的に送られるようになる。

４．想定通りにできたこと、できなかったこと

　上ではプロジェクトで実装できたものを述べてきた。具体的にはログイン機能とタスクの管理の機能ができている。しかし、まだできていない部分が残っており、目標を管理する機能がまだ実装できていない。この部分が非常に難しく、目標を入力するページを用意する必要がある。その画面の作成方法と、どうやって目標をインプットするかが問題点として残っている。１つの目標を達成するために、その目標の下位にある複数の目標を処理しなければならないのだが、それら下位の目標（タスク）の作成を自動的に発生させるか、もしくは全部手で入力するかが問題になった。自動的に発生させる方法の方が勿論難しい。手で入力する方法は目標とはいえず、ただのタスクの羅列と変わりない。しかし、今回は自動的に目標を発生させる部分ができなかった。確率を用いて目標ができるかどうかを判断する必要があったのだが、その機能を作成するには非常に時間がかかってしまうということが理由である。