

Lecpool On RoR

Summer Project, 2011

Programming Club

Science and Technology Council

IIT Kanpur

By: Pratik Agarwal

Sanchit Gupta

Ujjwal Singh

Mentor: Adarsh Jagannatha

I. Ruby On Rails

1. What is RoR ?

RoR stands for Ruby on Rails. Ruby is the programming language and Rails is the platform that supports that.

2. About Ruby On Rails:

Ruby on Rails was extracted by [David Heinemeier Hansson](#) from his work on [Basecamp](#), a project management tool by [37signals](#) (now a [web application](#) company).Hansson first released Ruby on Rails as open source in July 2004, but

did not share [commit](#) rights to the project until February 2009. In August 2006 the framework reached a milestone when [Apple](#) announced that it would ship Ruby on Rails with [Mac OS X v10.5 "Leopard"](#), which was released in October 2007.

Ruby on Rails version 2.3 was released on March 15, 2009. Major new developments in Ruby on Rails include templates, engines, [Rack](#) and nested model forms.

- Templates enable the developer to generate a skeleton application with custom [gems](#) and configurations.
- Engines let one reuse application pieces complete with routes, view paths and models.
- The [Rack](#) web server interface and Metal allow one to write optimized pieces of code that route around ActionController.

On December 23, 2008, [Merb](#), another web application framework was launched, and Ruby on Rails announced a commitment to work together. The Ruby on Rails team announced they would work with the Merb project to bring "the best ideas of Merb" into Ruby on Rails 3, ending the "unnecessary duplication" across both communities. Merb was merged with Rails as part of the Rails 3.0 release.

3. Technical Overview:

Like many web frameworks, Ruby on Rails uses the [Model-View-Controller](#) (MVC) architecture pattern to organize application programming.

Ruby on Rails includes tools that make common development tasks easier "out of the box", such as [scaffolding](#) that can automatically construct some of the models and views needed for a basic [website](#). Also included are [WEBrick](#), a simple Ruby web server that is distributed with Ruby, and [Rake](#), a build system, distributed as a [gem](#). Together with Ruby on Rails these tools provide a basic development environment.

Ruby on Rails relies on a [web server](#) to run it. [Mongrel](#) was generally preferred over WEBrick at the time of writing, but it can also be run by [Lighttpd](#), [Abyss](#), [Apache](#), [nginx](#) (either as a module - [Passenger](#) for example - or via [CGI](#), [FastCGI](#) or [mod_ruby](#)), and many others. From 2008 onwards, the Passenger web server replaced Mongrel as the most used web server for Ruby on Rails.

Ruby on Rails is also noteworthy for its extensive use of the [JavaScript](#) libraries, [Prototype](#) and [Script.aculo.us](#) for [Ajax](#). Ruby on Rails initially utilized lightweight [SOAP](#) for web services; this was later replaced by [RESTful web services](#). Ruby on Rails 3.0 uses a technique called [Unobtrusive JavaScript](#) to separate the functionality (or logic) from the structure of the web page, and [jQuery](#) is fully supported as a replacement for [Prototype](#).

Since version 2.0, Ruby on Rails by default offers both [HTML](#) and [XML](#) as output formats. The latter is the facility for RESTful web services.

Ruby on Rails 2.3 relies on [Ruby 1.8.6](#). Ruby on Rails 3.0 has been designed to work with Ruby 1.8.7, Ruby 1.9.2, and [JRuby 1.5.2+](#); earlier versions are not supported.

4. Framework Structure

Ruby on Rails is separated into various packages, namely [ActiveRecord](#) (an [object-relational mapping](#) system for database access), [ActiveResource](#) (provides web services), [ActionPack](#), [ActiveSupport](#) and [ActionMailer](#). Prior to version 2.0, Ruby on Rails also included the [Action Web Service](#) package that is now replaced by [Active Resource](#). Apart from standard packages, developers can make [plugins](#) to extend existing packages.

5. Deployment

Ruby on Rails is often installed using [RubyGems](#), a package manager^[19] which is included with current versions of Ruby. Many [Linux distributions](#) also support installation of Ruby on Rails and its dependencies through their native [package management system](#).

Ruby on Rails is typically deployed with a database server such as [MySQL](#) or [PostgreSQL](#), and a web server such as [Apache](#) running the [Phusion Passenger](#) module.

There are many Ruby on Rails hosting services such as [Heroku](#), [Engine Yard](#), and [Rails Playground](#).

6. Philosophy and Design

Ruby on Rails is intended to emphasize [Convention over Configuration](#) (CoC), and the rapid development principle of [Don't Repeat Yourself](#) (DRY).

"Convention over Configuration" means a developer only needs to specify unconventional aspects of the application. For example, if there is a class *Sale* in the model, the corresponding table in the database is called *sales* by default. It is only if one deviates from this convention, such as calling the table "products sold", that the developer needs to write code regarding these names. Generally, this leads to less code and less repetition.

"Don't repeat yourself" means that information is located in a single, unambiguous place. For example, using the [ActiveRecord](#) module of Rails, the developer does not need to specify database column names in class definitions. Instead, Ruby on Rails can retrieve this information from the database based on the class name.

II. Why RoR?

We preferred using RoR over Php and MySql because of its simplicity and its DRY (Don't Repeat Yourself) philosophy which allows us to create a web application in a short period of time.

Numerous RubyGems already available over the internet helps our cause further.

Using RoR we incorporated more features in our app with ease and in less amount of time.

III. Developing the application

1. Choice of Operating System and Tools Used:

Although we could have developed our application on Windows itself, we decided to use Linux as our OS because the Terminal is easily accessible in Linux and setting up the Rails server is much easier in Linux than in Windows.

Tools Used : Ruby 1.9.2 language library , Javascript , HTML and CSS

Rails 3.0.7 platform

WEbrick server

Sqlite3 Database

Sqlite Database Browser

Gedit

2. RubyGems Used :

Meta-Search : This gem was used to implement the search tool in our application.

Sqlite3 : This gem was used to set up the sqlite database.

Rmagick, Imagemagick and Carrierwave: These gems were used for the purpose of uploading images and cropping it on the fly.

Will-Paginate : This gem was used for implementing the pagination system in our pages.

Heroku: This gem was used for deploying the app on heroku.

Thumbs-Up : This gem was used to implement the voting system.

Gravatar-Image-Tag: This gem was used for the purpose of uploading pics on heroku as heroku supports only read-only filesystem and carrierwave needs to write in the filesystem in order to upload the image. So, we had to compromise on this issue.

3. Models- Views-Controllers:

Database Models:

a. Attachments Database (for lecture files):

This database has the following columns→

Id: For storing the id

Filename: For storing the filename

Content_type : For storing the type of file

Data: For storing the data in binary format

b. Comments Database(for comments on questions):

This database has the following columns→

Id: For storing the id

Content: For storing the comment

User_id: For storing the id of user who posted the comment

Conversation_id : For storing the id of the question on which the comment has been posted

c. Conversations Database (for questions on a forum)

This database has the following columns→

Id: For storing the id

Title: For storing the Question Title

Description: For storing the actual question

User_id: For storing the id of user who posted the question

Forum_id: For storing the id of forum on which the question was posted

d. Forums Database (for storing the forums):

This database has the following columns➔

Id: For storing the id

Title: For storing the Forum Title

Description: For storing the forum description

User_id: For storing the id of user who created the forum

e. Users Database:

This database has the following columns➔

Id: For storing the id

Name: For storing the User's Name

Email: For storing User's email

Encrypted_password: For storing the encrypted password

Salt : For storing the salt id

Rollno : For storing the User's Rollno

Department : For storing the User's department

Cn1-Cn7 : For storing the course names to which the user has subscribed

Image : For storing the image file name

Admin: To specify whether a User is admin or not

Activated: To keep track whether an account has been activated or not

f. Videos Database:

This database has the following columns➔

Id: For storing the id

Title : For storing the video title

Video_code : For storing the video's url

User_id: For storing the id of user who uploaded the video

g. Votes Database (For managing Likes/Dislikes):

Id: For storing the id

Vote: True/False

Voteable_id : For storing the id of voted entity

Voteable_type: For storing the type of entity voted on i.e.

Attachment/Video

Voter_id : For storing the voter's id

Voter_type : For storing the voter type (here users)

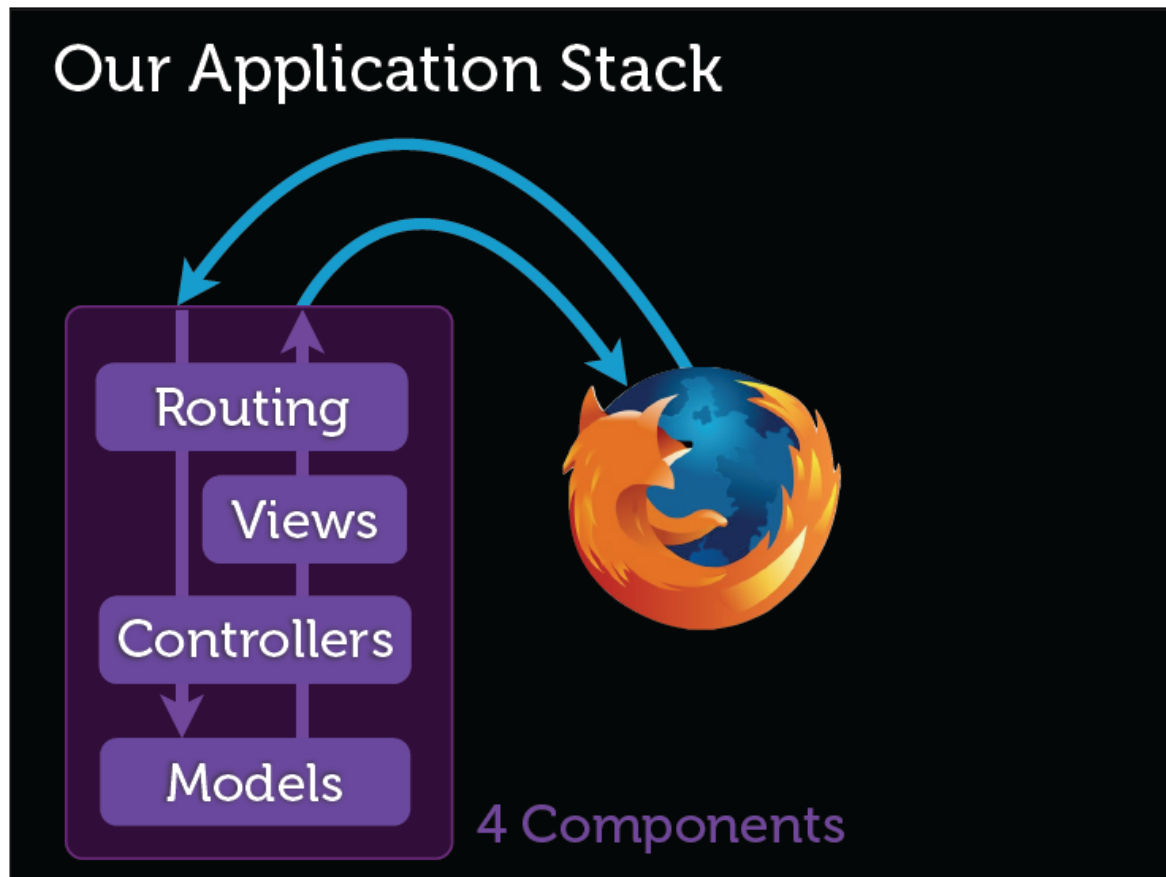
Controllers & Views :

Each Database mentioned above has a controller(details of which can't be discussed here) and a corresponding list of views.

Filesystem: It contains all the necessary files of the application. Moreover, the images uploaded through carrierwave gem also gets stored in the filesystem by default, however we can store it in Amazon S3, Rackspace etc.

4. How does MVC work ?

The following image is self explanatory, but still let me elaborate. When a User submits a request in a browser, it goes to a controller, the controller then fetches data from a model and then it renders a view. As simple as that.



5. Time Taken for developing the basic app : Approximately 30-35 days

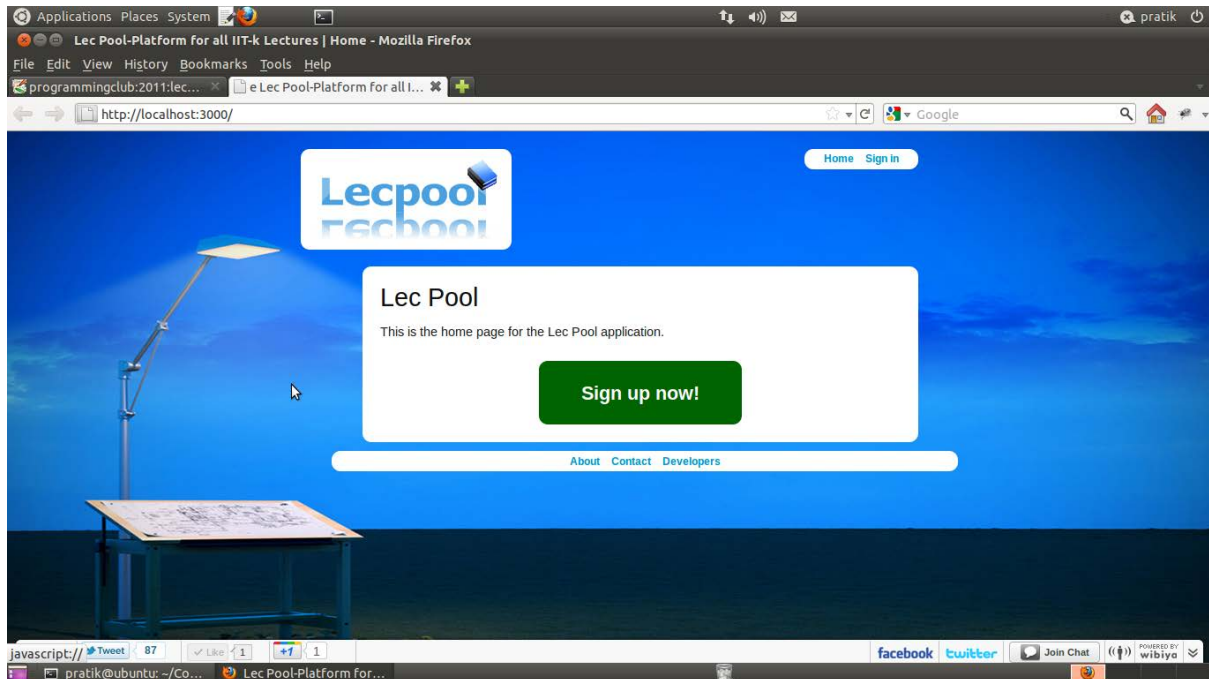
IV. Features of Lecpool

- An authentication system requiring email confirmation on Sign Up.
- Provision for Users to upload their profile pic.
- Provision for Users to upload their lecture notes in any format.
- Provision for Users to upload educational videos via Youtube.
- A Likes-dislikes style voting system.
- An Efficient Search Tool to help users find the lecture notes.
- A Forum where users can discuss their problems/issues.

- A Chat-client where Users can chat via their Facebook/Twitter/Myspace/Yahoo accounts. (Wibiya toolbar)

V. Screenshots Of the final app

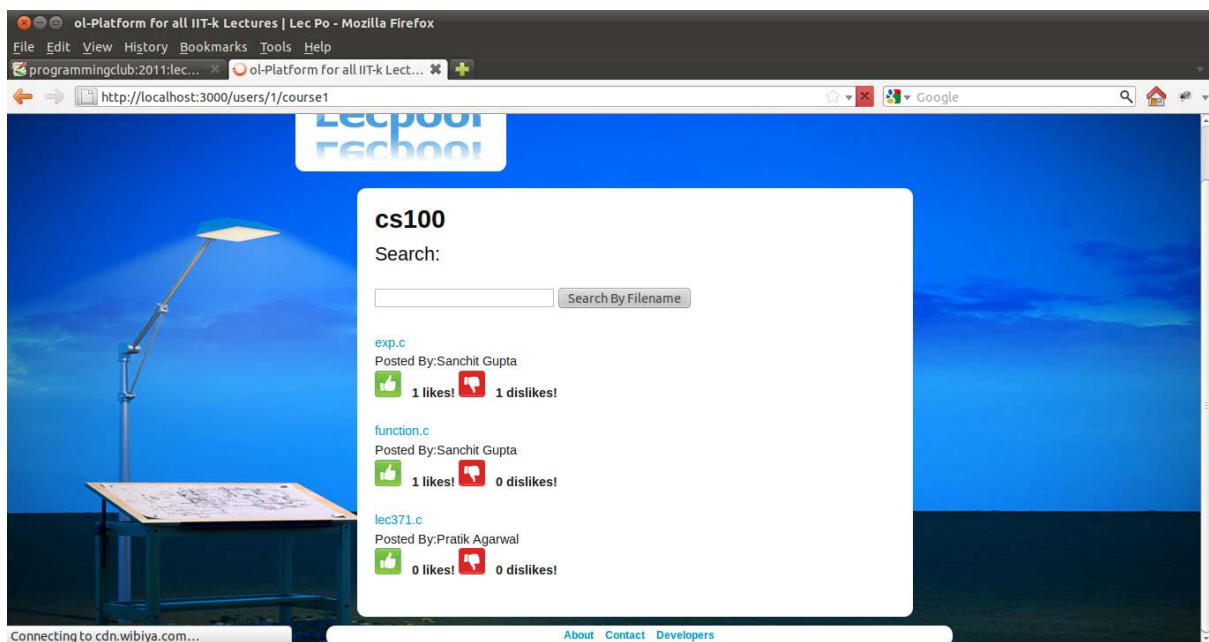
1. This is the Homepage of Lecpool



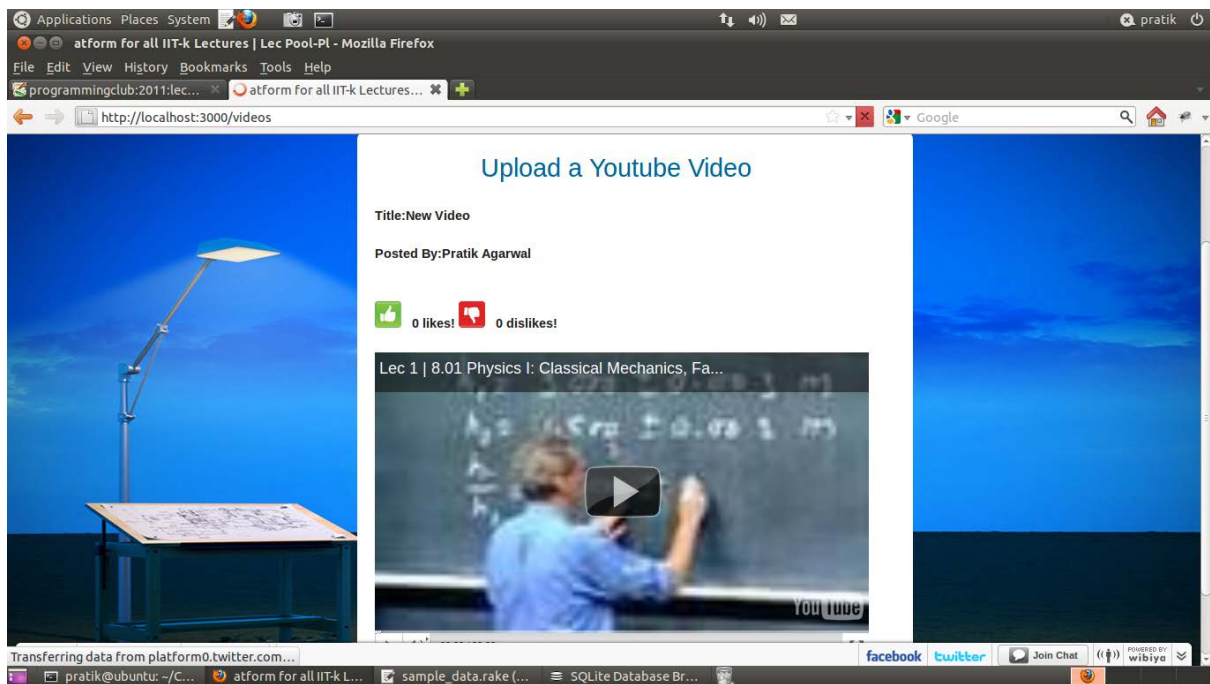
2. This is the User Profile page



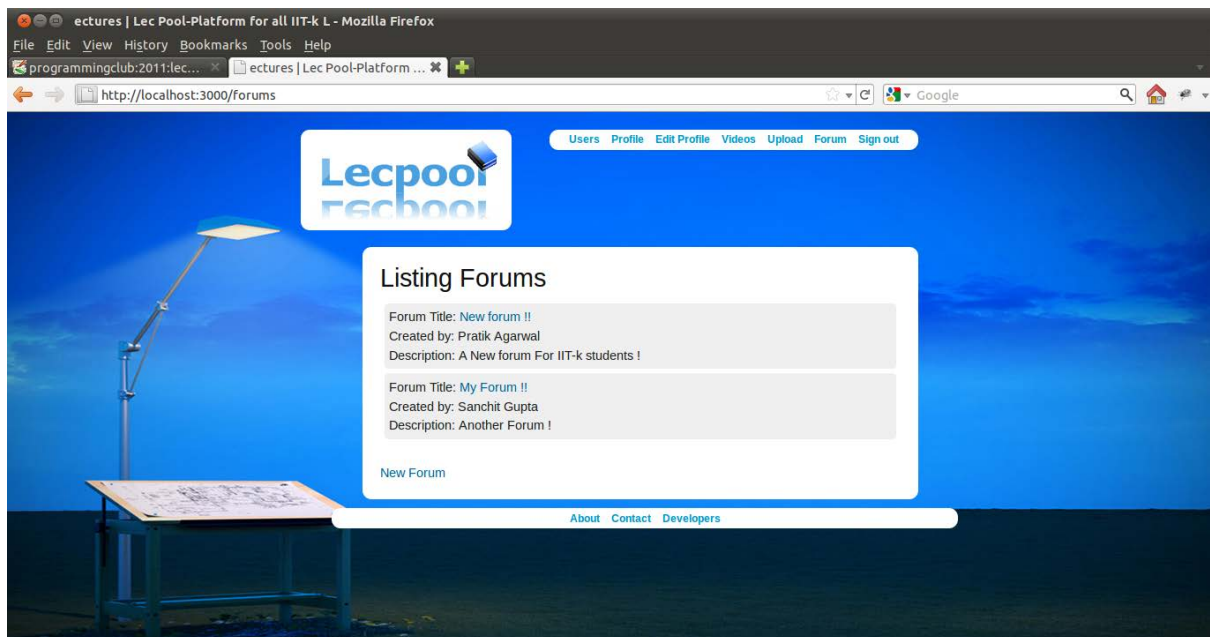
3. This is a Course Page displaying the lecture files



4. This is the videos upload page where Users can upload educational videos



5. This a Forums page



V. Scope For Improvement

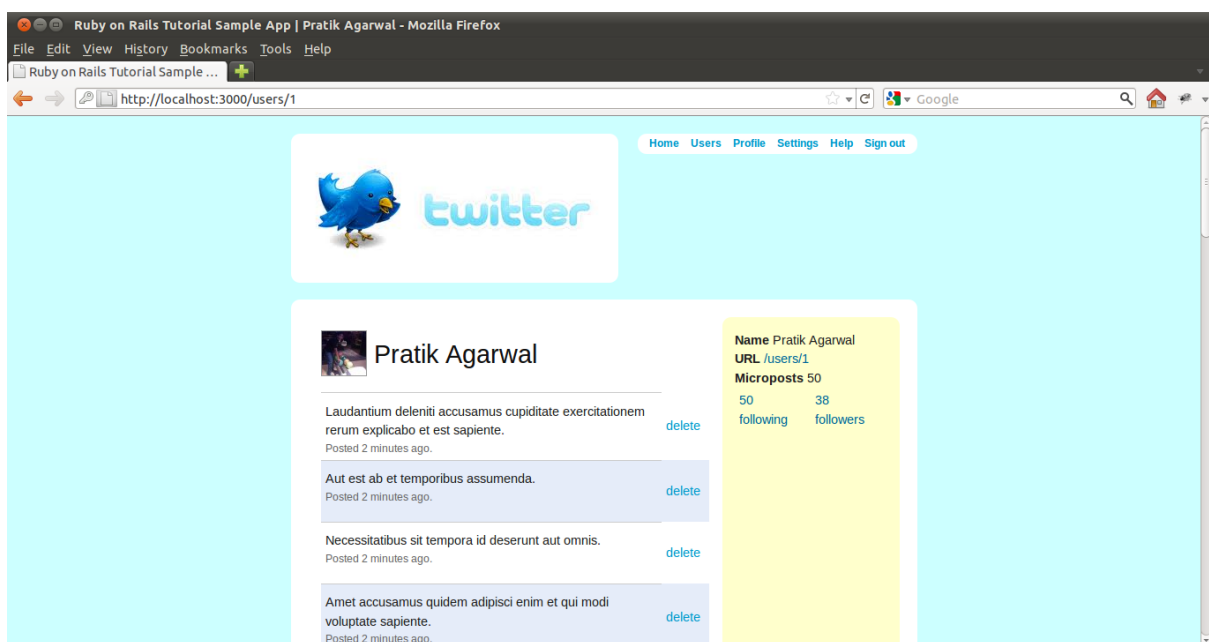
- Implementing a live feed-notifications system, to keep the users updated.

- Implementing a “Request Help from a Friend” system, where users can ask their specific doubts to a particular person.
- Implementing a Ranking system for users, where the Users whose answers are liked the most gets the highest rank.

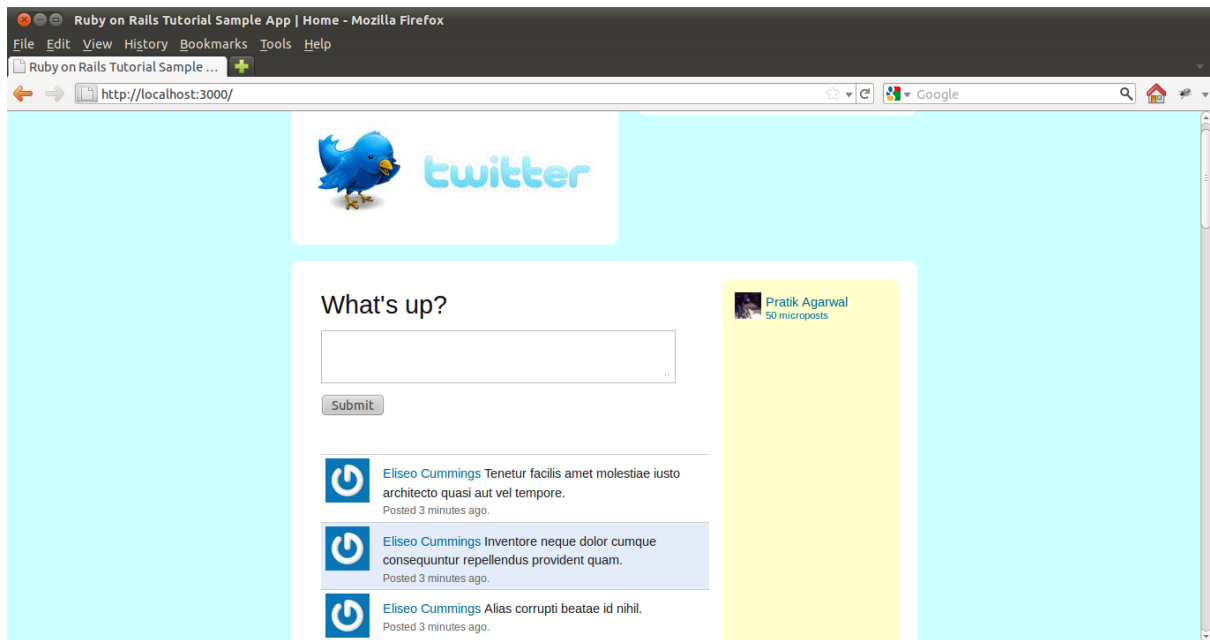
VI. By-Products

In the process of learning Ruby on Rails, we created an app resembling “Twitter”. Some of the screenshots of this app can be seen below:

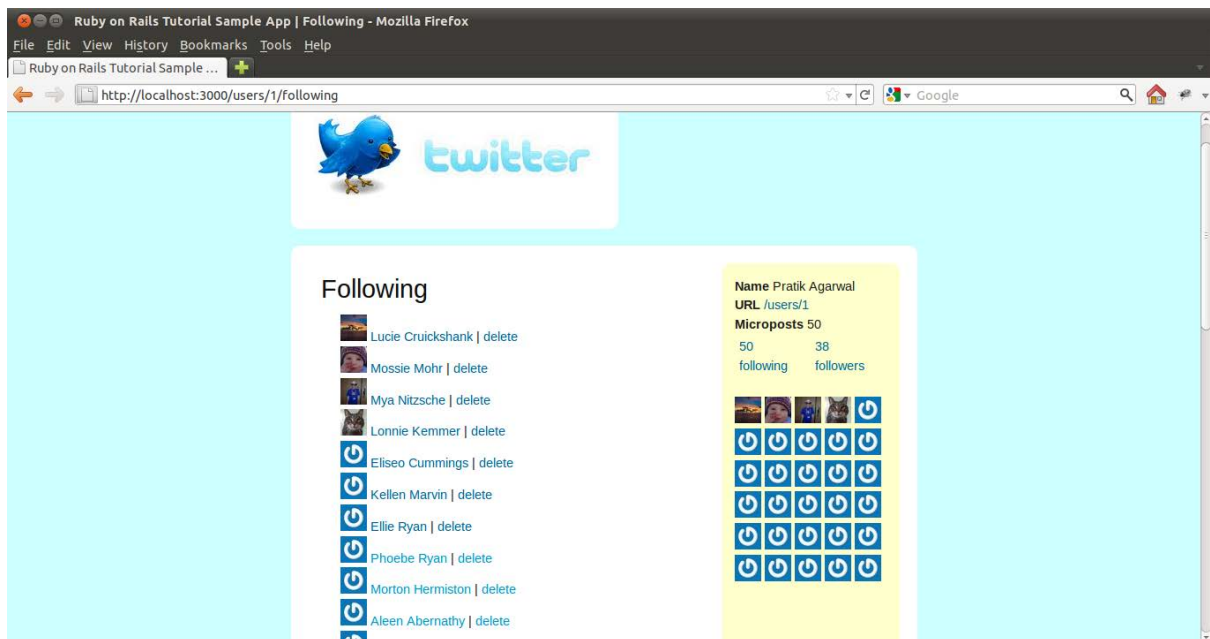
1. The User Profile Page displaying the User’s tweets



2. The User homepage displaying the tweets of all other Users



3. Page showing User's Followings



VII. References

- Ruby basics : <http://tryruby.org>

- Rails basics : <http://railsforzombies.org>
- Ruby On Rails Tutorial by Michael Hartl
- Answers to common problems: <http://stackoverflow.com>
- Plugins : <http://github.com>
- Google

VIII. Acknowledgements

- Co-ordinators of Programming Club IIT-K
- Sri Raj Dutt V Badam (Y7 , CSE)
- SnT Council IIT-K

IX. Relevant Links

- Heroku Link : <http://lecpool-iitk.herokuapp.com>
- Github Link : <https://github.com/aquaprats/lecpoolf>

