Cloud Calendar
COMP30520 Cloud Project

Evan O'Keeffe

10324289

UCD School of Computer Science and Informatics
College of Science
University College Dublin

May 2, 2014

1: Introduction

The aim of this project was to create an online cloud based calendar. This online calendar such as Google Calendar is a time-management web application. It allows users to create new events, edit or delete existing events, and search for events. Indeed, it also allows users to share their calendars. In this project, the students are asked to develop an online calendar with the following operations:

1. Create new events
2. Edit or delete existing events
3. Search for events
4. Time slots booking
5. Share calendar

This online calendar can be stored in cloud-based storage. We were allowed to use any web development tools to develop this online calendar.

For the cloud-based storage we chose the openly free OpenShift RedHat Cloud provider. They provide IAAS,PAAS services. They also provide a domain name and basic connection bandwidth. They do not charge you in anyway for the free service.

2: **Database Design**

The database was a very simple design. It contains three tables, User, Event, Permissions. The relationships are User-¿Events which is N-¿N. This is implemented using the Permissions table. Permissions uses the Users ID and Event ID to allow users to see events that are shared with them. An event hook is attached once a calendar is shared, this means when a calendar has been shared that new events are added , the users being shared with is updated to allow them to see these events.

2.0.1 User

- ID: Unique , Incremented integer
- Username: Unique UTF8 String , the users username
- password: UTF8 String, Encrypted with Bcrypt
- last logged: DateTime , Last time user was logged in at

2.0.2 Event

- ID: Unique , Incremented integer
- Title: Secondary Key for this table, UTF8 String, Event Name
- Description:
- Start: Secondary Key for this table, DateTime object, Event start date and time
- End: DateTime object, Event end date and time
- All Day: Boolean value , does the event last all day ?
- URL: UTF8 String , the events URL address
- creator: Integer Back reference to Permissions table, ID of the user that made this
- users: List Object , Foreign Key to Permissions table
- created: DateTime object, when Event was created
- last edited: DateTime object, last time event was edited

2.0.3 Permissions

- user˙id: Foreign key to User Table attribute ID
- event˙id: Foreign key to Event Table attribute ID

3: **Software Used**

This web application was developed using the Python programming language. We chose this as it is highly efficient for building applications in a small amount of time. We chose the Pyramid Web Application Framework for building the website back end. It creates as a Web Server Gateway Interface (WSGI) application. Pyramid has great documentation and is based off the highly successful and implemented Pylons project. It has many plugins and add-ons that integrate easily with the framework.

For the Pyramid design we used for Mako templates that allowed us to create dynamic AJAX web pages that would change whether the user had authenticated themselves, this authentication is then saved in the request object contained in the users browser. This request is encrypted to prevent people trying to log in as other users by obtaining the request object.

For the security we used ACL authentication, for the cookie's we used the basic authenitcation that comes with Pyramid which implements SHA512. For the users password we used Bcrypt, a modification of unix's crpyto library for python that uses machine strength to progressively increase the logarithmic difficulty of the password. It runs slower than other libraries but is the current most secure library for encrypting in python applications.

For the database we used MySQL with SQLALchemy, an object to relation mapping library for python. It turns SQL tables into Python classes and back to allow for easier development with SQL engines. For the transactions we used Zope transaction manager, this would check database consistency when adding,editing and deleting entries in the database to prevent race conditions and deadlocks.

For the forms we used WTForms , a python library that creates nice web forms from python SQLAlchemy tables. It integrates nicely with Mako and has plenty of documentation.

For the front end user interface we used Mako HTML5, CSS, Javascript, Jquery building the UI. For the calendar creation and maintenance we used FullCalendar.js which was then linked using JavaScript to the python back-end server functions.

4: Finished Project Conclusions

Using OpenShift we successfully implemented all tasks set out. The end product works quite nicely. A URL for the website we've set it up on is available @ CloudCalendar.

We were very happy with how it turned out. Below we've included screenshots of a few screens to show how it looks. I'm releasing a tutorial to show others how to make this using python pyramid and releasing the code freely online under a BSD license.

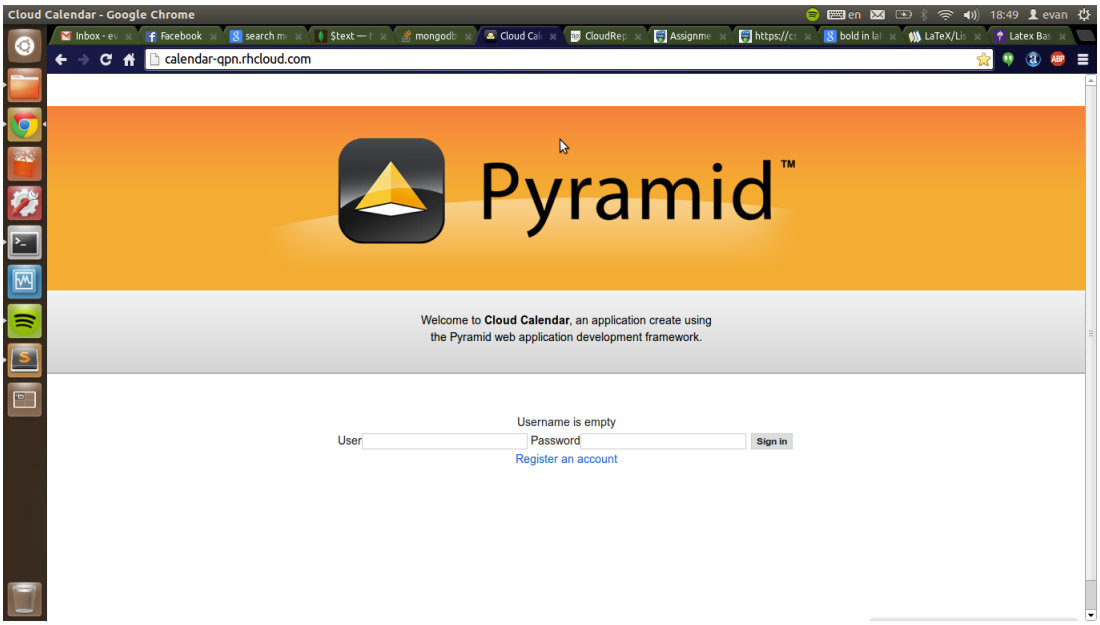

Figure 4.1: Website home when User is not logged in

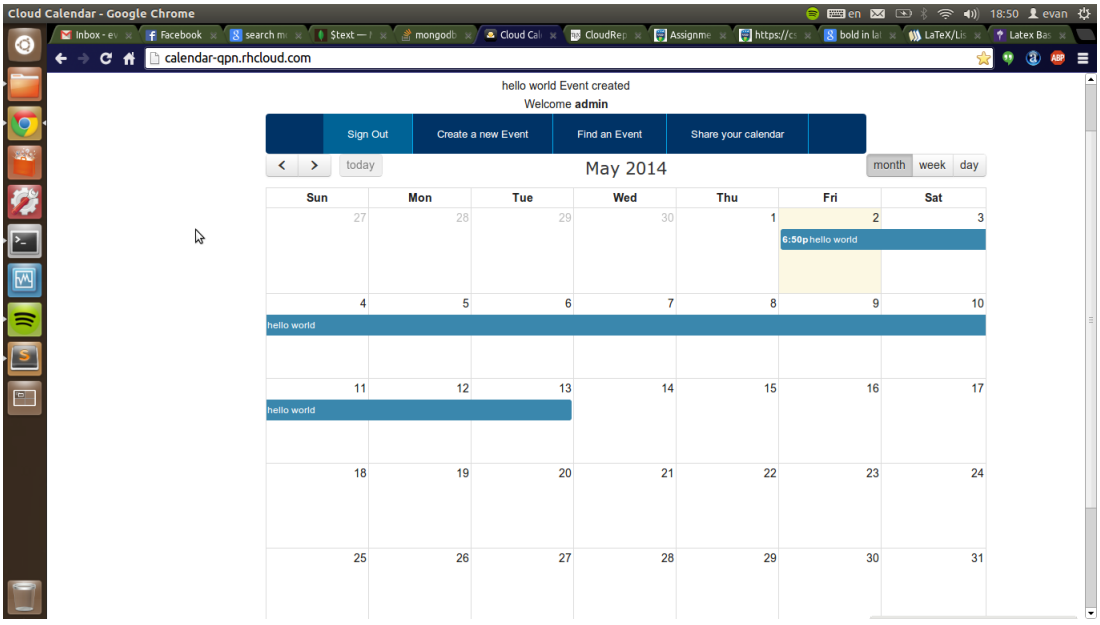

Figure 4.2: Website home when User is logged in

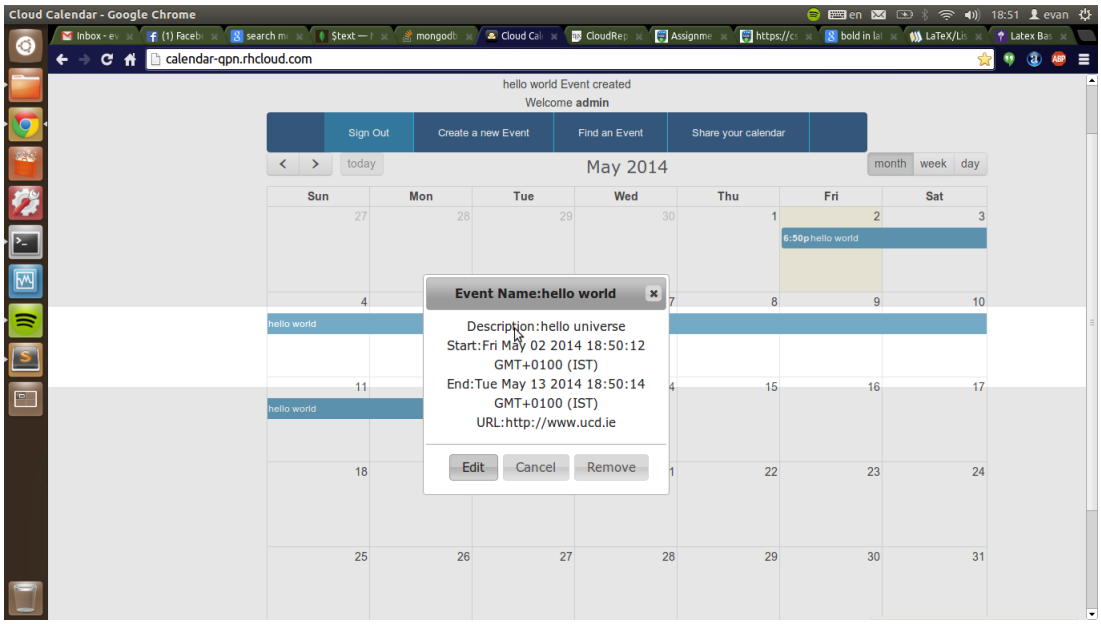

Figure 4.3: Modal that pops up on click event

5: All Modules and Libraries Used

5.0.4 Cloud provider and Services used

- OpenShift
- Database: MySQL
- Gear Scaling: Scale services up or down to users loads
- Domain and URL management

5.0.5 Python

- pyramid
- mako
- pyramid˙mako
- pyramid˙chameleon
- pyramid˙persona
- pyramid˙debugtoolbar
- repoze.tm2¿=1.0b1
- pyramid˙tm
- SQLAlchemy
- transaction
- zope.sqlalchemy
- waitress
- WebError
- apex
- deform
- wtforms
- webhelpers
- tw2.core

- tw2.forms
- tw2.dynforms
- tw2.sqla
- tw2.jqplugins.jqgrid
- tw2.jqplugins.fullcalendar

5.0.6 Javascript/Jquery

- FullCalendar.js
- datetimepicker

5.0.7 CSS

- BlueMineral menu.css