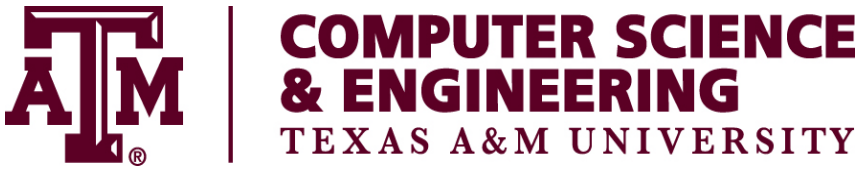
CSCE 606 -Software Engineering, Project Report

Clock Timer for scheduling and synchronizing Radio Shows

by

Naveen George Thomas, Sanghita Bandyopadhyay, Manisha Tripathy, Ranbir Das, Sangeeta Panigrahy and Atin Ruia



1. Two paragraph summary of the project ***as implemented***, including the main customer need and how the application meets it, including who the stakeholders are. This will contrast to what you wrote in Iteration 0.

   Project aimed at providing the customer: Dennis Macha, a radio show host at RedC Radio Station, with a cross platform web application for keeping track of radio show schedules and managing them efficiently. The application developed was intended to replace an in-efficient, platform dependent and unhandy application which was currently being used by the customer for the same purpose. User functionality was provided to enable the re-use of the application across multiple radio show stations. Application data mainly consisted of information regarding radio shows organized as different segments across various timings in a week. An alert page displayed to the user consisted of a graphical interface intimating the user about the radio show segments scheduled for the day and a show timer synchronized to an user configured time zone(CST, PST, MST, EST). Segments were classified as past, current and upcoming events based on their start/end timings. As the data to be managed can often be large, query functionalities were provided to the user for searching a particular show and its segments based on its name/scheduled_days. User was provided with an intuitive configuration page for entering the schedule of radio shows and their segments and editing them as and when necessary. Access to any users data is secured via an user authentication functionality. Admin functionality was also provided to take care of user management details. Settings corresponding to the scheduler/alert system such as flash alert time intervals, time zone server synchronization were made user configurable as per customer feedback.

2. Description of ***all*** user stories. For each story, explain how many points you gave it, explain the implementation status, including those that did not get implemented. Discuss changes to each story as they went. Show lo-fi UI mockups/storyboards you created, and then the screen shots, as needed to explain.

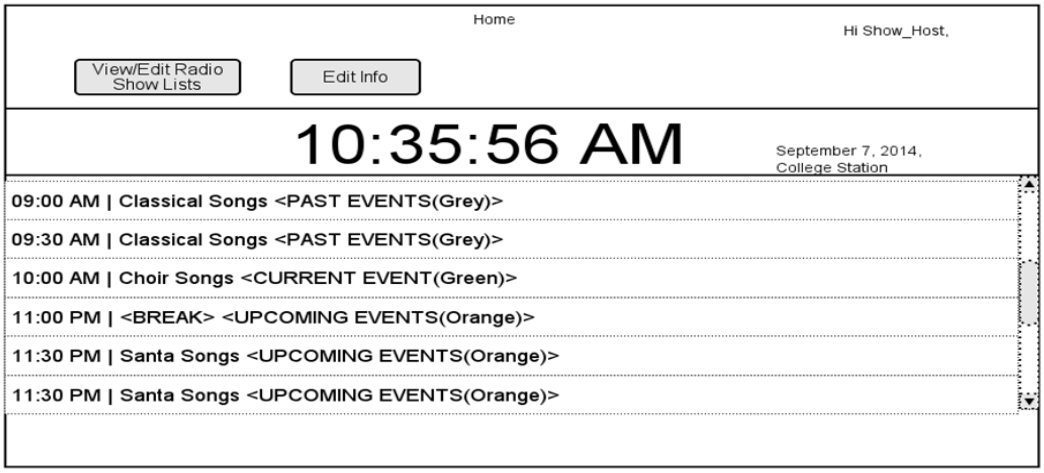
   User Stories are as follows :-

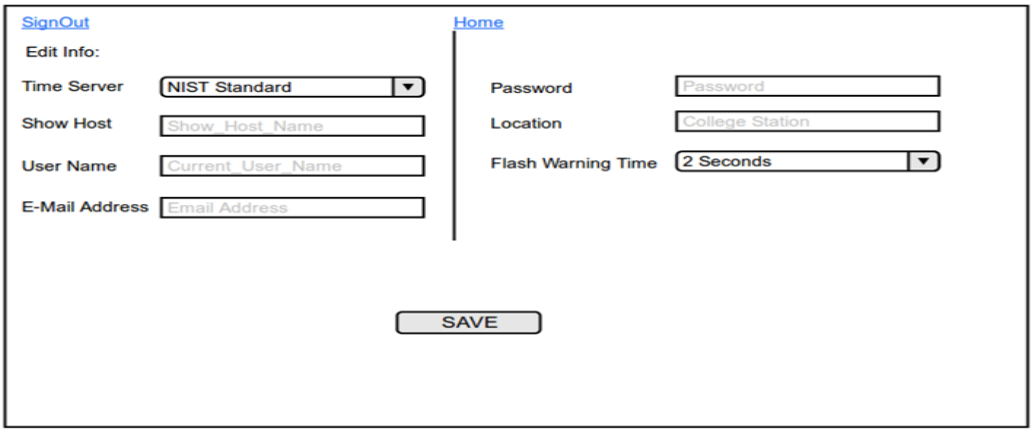
Serial No.	User Story	Points	Implementation Status	Description
1.	Flash alert for event expiration	2	Implemented	The purpose of this functionality was to provide a flash alert on the timer for the current show when the remaining time for the show equals the flash time provided by the user.
2.	Customizable hourly segment / daily planner	2	Implemented	The purpose of this functionality was to enable the host to add/edit/delete show segments

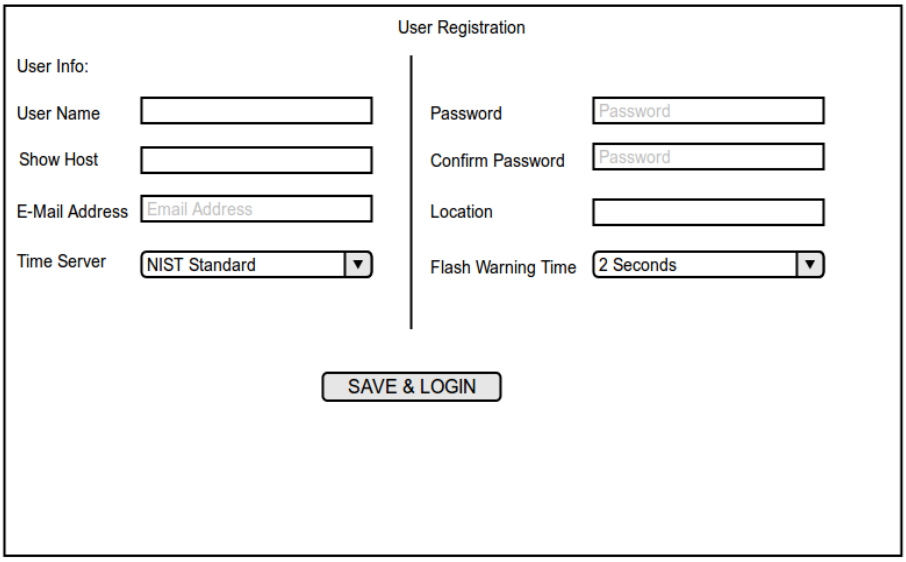
				in-order to create the daily schedule for the radio station.
3.	Sync show time clock with internet time server	2	Implemented	Radio shows are scheduled according to the local time server and it is required to sync them with the host radio station. Our functionality synced the show timer with the chosen time server. This would enable radio stations to host shows synced with different time servers. Our application provides four time servers to choose from. They are:- a. Pacific Time b. Mountain Time c. Central Time d. Eastern Time
4.	Add current and upcoming event description	2	Implemented	We provided a dynamic list of upcoming as well as elapsed shows. This was intended to help radio show hosts to keep track of the progress of events scheduled for that day.
5.	Add notification	2	Implemented	We have provided two options to view/edit the show list. Either the user can select to view shows by day of the week or by name of the show. If user chooses either of the options, they are notified about all the existing shows. They also have the option to edit/delete any of these shows.
6.	Add Authorized Access only	2	Implemented	In-order to access our application we direct the user to signup first. Only users with valid user credentials are allowed to login. We have provided separate authorization for admin access where the authorised personnel can manage all the user accounts.

7.	Make the app cross platform	1	Implemented	Our client wanted the application to be cross-platform. Since we have followed the REST-principles for developing our application.
8.	Resizing app window	1	Implemented	We used BootStrap themes with webkit layout engine to provide dynamic resizing of the content divs.

UI Mock-ups







3. List who held each team role, e.g. scrum master, product owner. Describe any changes in roles during the project.

Team Roles:

a. Product Owner : Sangeeta Panigrahy
b. Scrum Master : Ranbir Das
c. Team Member : Atin Ruia
d. Team Member : Naveen George Thomas
e. Team Member : Manisha Tripathy
f. Team Member : Sanghita Bandyopadhyay

4. List the scrum iteration schedule (e.g. start date of each scrum). For each iteration, explain what was accomplished and points completed.

Scrum Iteration Schedule

Iteration	Start Date	Accomplishment	Points Completed
1.	10/23/14	User stories and mockups	8/8
2.	11/1/14	Application functionality update	5/5
3.	11/8/2014	First prototype presented	1/1
4.	12/1/2014	Graphics update and application hosted	4/4

List of customer meeting dates, and description of what happened at the meetings, e.g. what software did you demo.

Customer Meeting Dates

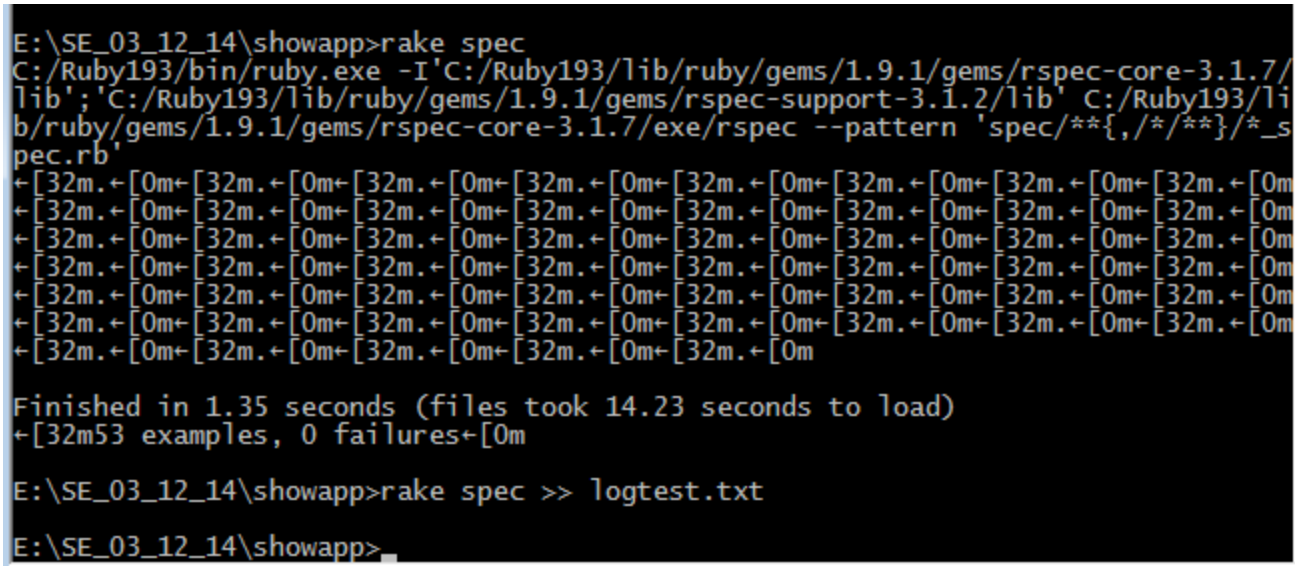
	Date of Meeting	Agenda	Demo
1.	10/16/2014	Initial meeting,discussing and collecting requirements and details	NA
2.	10/19/2014	Two-minute video of customer describing requirements	NA
3.	11/07/2014	Discussing UI Mock Ups with the client and getting relevant inputs.	UI Mock Ups
4.	12/02/2014	Demonstrating the application with the current development status and getting feedback.	Application Demo
5.	12/08/2014	Demonstrating the updated application after Incorporating the changes requested in the previous meeting and seeking feedback.	Application Demo

5. Explain your BDD/TDD process, and any benefits/problems from it.

Test-driven development (TDD) is a software development process that uses very short development cycles. Behaviour Driven Development is another software development process based on test-driven development (TDD). The two gems provided by rails namely, RSpec and Cucumber are used for pursuing BDD and TDD. We began by identifying the main features of our application from the user-stories. Using these we created several scenarios per feature to cover all possible acceptance test scenarios.
Once we had all our scenarios defined we proceeded to develop functionalities per scenario.For this we isolated all functionalities pertaining to each user story as a RSpec class. RSpec,enabled us to pursue Behaviour-Driven Development as well as Test-Driven Development.

Commands used to perform RSpec testing:-
>rake db:migrate RAILS_ENV=test
>rake spec
Result:-
Finished in 1.21 seconds (files took 13.6 seconds to load)
53 examples, 0 failures

Snapshot of final RSpec post-development with all examples passed.



A sample RSpec test-case used by our application is as follows:-

```
describe "GET show" do
  it "assigns the requested radio_show as @radio_show" do
    radio_show = RadioShow.create! valid_attributes
    get :show, {:id => radio_show.to_param}, valid_session
```

```
    expect(assigns(:radio_show)).to eq(radio_show)
    end
end
```

Benefits from BDD/TDD:-

a. It helped us to get the most important functionalities sorted out at the onset and get a comprehensive model of the application.
b. It also helped in identifying most of the user paths and implement appropriate methods to handle them.
c. Enabled us to develop the application from the user point of view and streamline the functionalities according to the needs of the user.

Disadvantages from BDD/TDD:-

a. A lot of extra time was spent to develop the test cases which mostly focused on the functionalities of the application rather than on its performance.

6. Discuss your configuration management approach. Did you need to do any spikes? How many branches and releases did you have?

We used the Git repository for configuration management. At the onset, we pushed the initial configuration of our project into the remote Git repository. It could be accessed by all members of the project. As we started working on the user stories and developing scenarios, we gradually updated the Git repository. Before pushing in any changes into the Git repository we did rigorous testing of the code so that the code in the main branch has all the valid changes. Each member branch had its own version of the code with their respective changes.

In agile software development, a spike is a story that cannot be estimated until a development team runs a timeboxed investigation. We did not face the necessity of conducting a spike since the user stories were quite well defined.

7. Discuss any issues you had in the production release process to Heroku.

We used Heroku to deploy our final application. Heroku internally maintains a git repository and requires one to use PostgreSQL or MySQL as the production database. Since we were initially using SQLite as our development, test and production databases, there were many configuration changes required in our rails application for it to just load onto Heroku repository. Besides that, we initially faced some issues with the graphics of our application once loaded into Heroku. It required us to precompile the assets folder and do some configuration changes in the production.rb file for it to function correctly.

8. Describe your implementation environment – homebrew, VirtualBox, AWS and any issues with it.

We implemented our application using VirtualBox.It runs on Windows, Linux, Macintosh, and Solaris hosts.Since it suports so many guest operating systems,it made the task of synchronising the work of different team members working on different operating systems,easier.
Issues faced:

a. There were some hiccups in obtaining a stable version of VirtualBox compatible with both Windows and Linux.
b. It did not give us any guarentee of performance across platforms.

9. Discuss the other tools you used, e.g. Git, Pivotal, CodeClimate, etc., and what you learned from them.

Following are the list of tools used:-

a. GitHub repository: We utilized the public GitHub repository to manage all the versions of our code as well as manage the code changes made by different developers parallely. It helped us in maintaining a stable version of the code. The GitHub repository for our project is populated and can be accessed at https://github.com/spsangeeta/ShowClockTimer

b. Pivotal Tracker: Pivotal tracker was used to record every iteration i.e. the code changes and new inclusions. It helped in keeping a track of the evolution process of our code which made the task of backtracking easier.

10. Link to 2-minute video interview.

A video interview depicting the customer requirements for this project was done. It is posted and can be accessed at https://vimeo.com/110113222

11. Link to demo.

The final application is hosted at https://radioshow.herokuapp.com
Please find the link to the application tutorial at https://vimeo.com/114628827

12. Link to user manual

As requested by Mr. Dennis Macha (customer) we have prepared a user manual to help the user to navigate through the application. It can be accessed at https://drive.google.com/file/d/0B75F09P6zBNgcXgwM0M4cUxyWFE/view?usp=sharing