**Scheduler**

*Miguel Castro*

*Ryan Tolentino*

Date Submitted: 05/05/2018

**Introduction**

Scheduling can be difficult especially with a busy schedule. Scheduler is a web based application that makes scheduling a breeze. It is a simple to use application that mitigates complexity and gets right to the point. In a simple explanation, the application takes two individuals, and notates specific information. Then, an activity is made between parties and the receiver either approves or denies the activity. Both individuals will get an email that highlight information of the requestor and the appropriate activity. This final paper will go over the team’s progress and development of the application.

# **1. Problem Statement**

The team must perform a thorough investigation on an idea between the group. Afterwards, the team must implement the idea and create a working sample. The team then must select an advisor that corresponds to the desired topic. The team must create an initial proposal and present it the advisor and instructor. The team must then document progress and the development process of the project. The team must continually work on the project until the deadline. The team must update the advisor and class instructor on any changes within the development process. Finally, the team must create a final paper and presentation showcasing to the Computer Science department the finished project.

**2. Background**

Web2py is a fairly new web design framework that utilizes Python. Web2py was started by Massimo Di Pierro in 2007 with the goal of making web development easier, faster, and more secure. The framework has three goals: ease of use, rapid development, and security. The main driver for why Web2py is so intuitive because of the model view controller. Web2py separates the data representation (the model) from the data presentation (the view) and also the application logic and workflow (the controller).

**2.1 Web2py Compared to Other Web Frameworks**

Web2py is a unique framework compared to others. It is very similar to more widely used applications such as Django and Ruby on Rails. However, Web2py provides a visual interface making it very easy to get an application up and running in no time. However, since it does provide a visual interface, web2py lacks functionality compared to scratch web frameworks.

**2.2 Model**

The model is used in Web2py to store our data which uses an SQL database to store our data. We created a model that stores the information for the scheduled event (Figure 1). This is populated by the scheduled events and is updated when the receiver either declines or accepts. However, the model can only be viewed from the admin side of the web application (Figure 2).



Figure 1. Model for Event

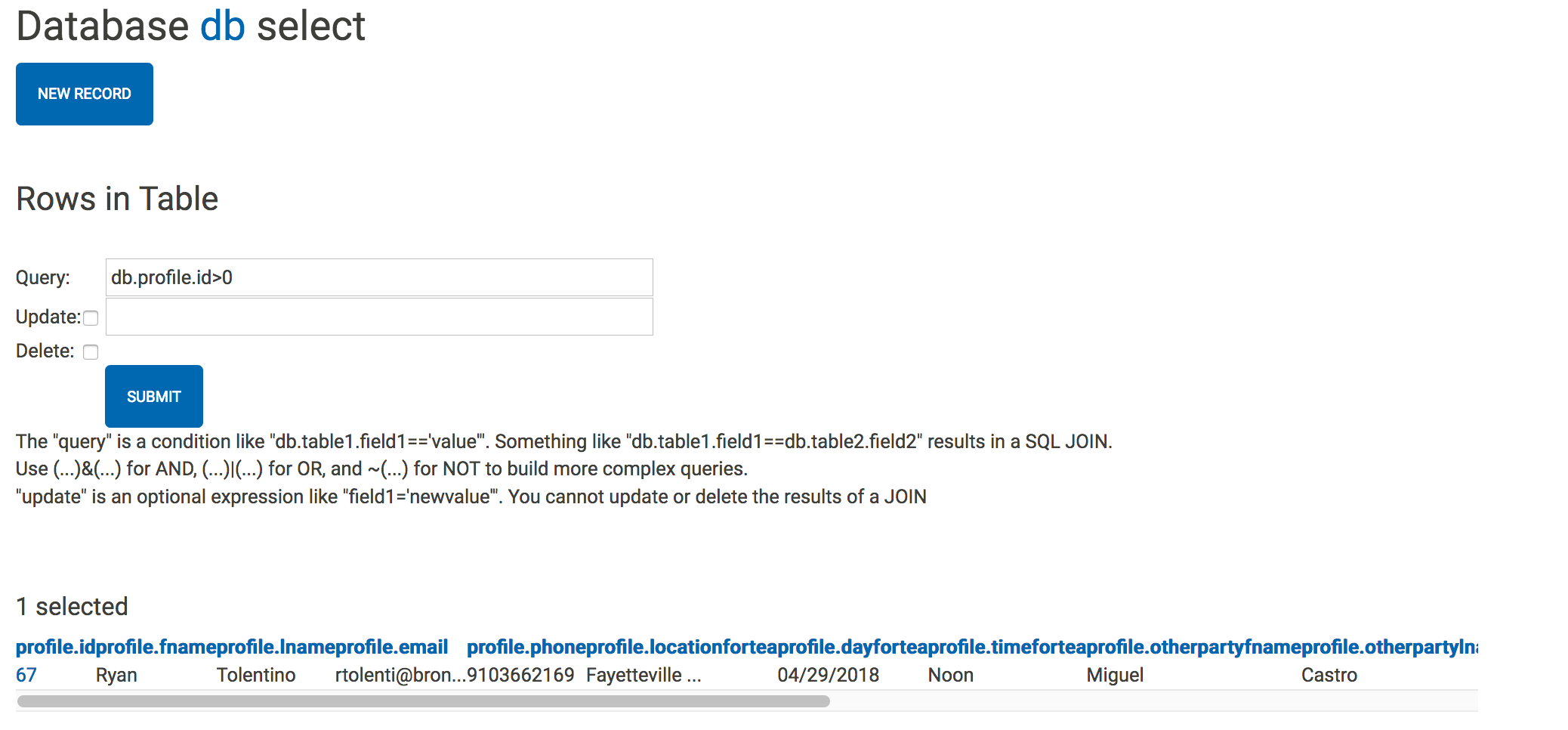


Figure 2. Admin View

**2.3 View**

The view is the web applications visual appearance. The view is attached to the model and the controller and gets the data to present. The views utilize Hypertext Markup Language (HTML) Cascading Style Sheets (CSS) and some Javascript (JS) to make the application more interactive.

**2.4 Controller**

The controller is the connector between the person using the application and the model. The controller allows the user to provide input. This controller is responsible for controlling the overall logic flow of the application. For example, when the form is submitted, the controller is utilized to send an email to the provided user input.

**3. Assumptions**

As a team, we made assumption to bound the scope of our project. With this assumptions, we go through the design process of our program and later revisit the assumptions to calculate any risk assessments. First, we assumed that both parties had @broncos.uncfsu.edu ending email addresses. In the event that an email that did not contain the proper email address, the email would not be sent. The team also assumed that the requestor had the contact information for the requestee. If the requestor did not have those details than it would be impossible for the application to work. Finally, we assumed both parties have access to an internet connection. The application, thus far, only will work with an active connection.

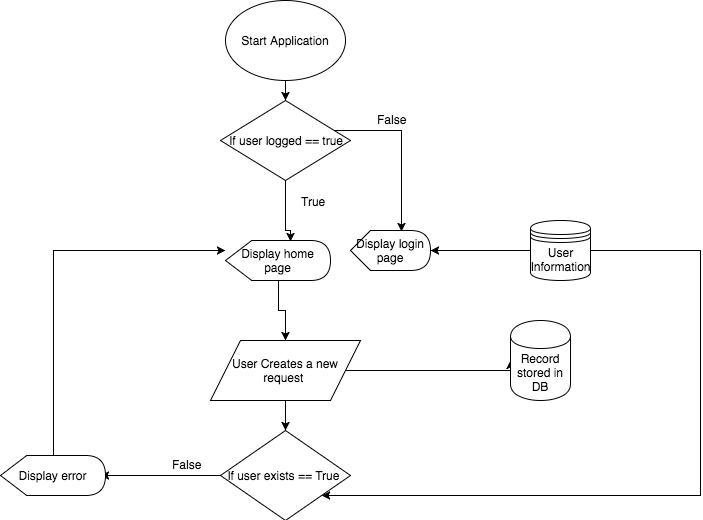
**4. Techniques and Tools**

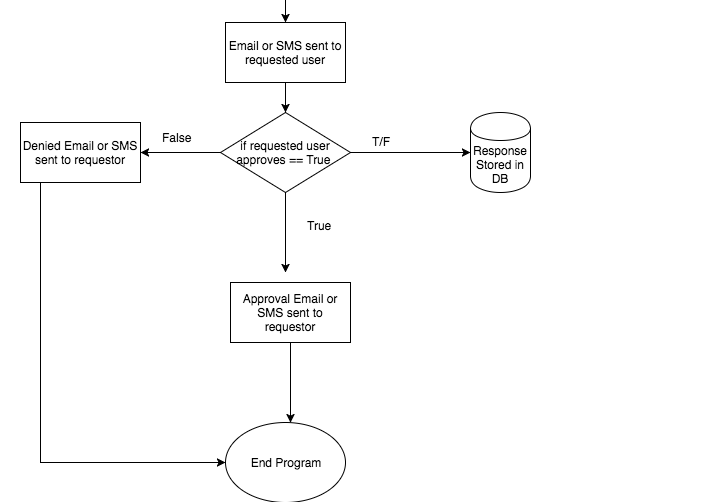
**4.1 Web2py**Web2py is a free, open-source web framework for secure database web application. It is a built for rapid development of fast, scalable, secure and portable database-driven web applications. The application uses Python and includes a database abstraction layer, therefore it knows how to generate SQL by itself. **4.2 SQLDB**A database is used for a structured set of data held in the computer. In our application we will be using an SQLDB. Web2py supports SQLDB as is the primary tool to store user information as well as the requestee.  
**4.3** **HTML, CSS, Javascript**To design the view of our web application, Hypertext Markup Language (HTML), Cascading Style Sheet (CSS), and Javascript makes the application interactive. These three tools go hand in hand with the Web2py framework and gets translated into what the user will see once they visit the application page. We were able to use online resources to help us build the website.

**4.4 Mail Server/SMTP**  
This web application utilizes a single mail server to send and receive emails. The web server uses the @broncos.uncfsu.edu mailing address to connect the end user together. The mail server uses the Simple Mail Transfer Protocol (SMTP) to send and receive emails.  
**4.5 Calendar Application Program Interface (API)**  
Web2py allows the use of plugins and APIs. We decided to take advantage of this by utilizing a basic calendar API for ease of use. The calendar API is used to figure the date of the specified activity,

**5. Problem Solution**

For our final project, we worked with Dr. Longfei Wu as our advisor and created a web based application. Our application has gone through several iterations. The final product is an application we call Scheduler. The web address is <https://web.uncfsu.edu/tea/>.

****

****

Upon completion of the flow diagram we then started coding the web application. First, we created a new application using the web2py admin interface. Web2py allows for the developer to utilize a mailing server to send emails. Second we, provided the IP address and name for mail server. We then started to craft the database and we decided to create on giant DB that stored the users information (Figure 1). After the completion of the model we needed to create a view for the index page. Therefore, we utilized a template from w3schools.com to create a nice and attractive landing page. However we realized that you can not just have pure HTML alone we needed to import a CSS file also provided by W3schools.com. Upon completion we linked the home page to a second view which was the actual application for the event. After completion of the second view we then started crafting the controller of the application which took the inputs provided in the view and stored them in the database. In the controller after the form was submitted they would be brought to a third view and an email would be sent to the provided email address of approval. The other party would then open up their outlook email box and it would contain an email from out mail server scheduler@tolentinodev.com . The other party would follow the link and then view the link sent to them. In the controller for ever request a new view is created based on the id of the request. The other party would then click deny or approve and based on the input the controller would either send a rejection email or approval email. If the event was approved the controller would then run a function called grid\_update() which would update the grid and active a different part of the controller sending an email to the requestor detailing the updates of the event. Lastly, we created an Admin view that allowed Miguel and I to see and delete requests stored in the database. For more details for our problem solution please try out our application, <https://web.uncfsu.edu/tea/>.

**6. Risk Assessment**

As we completed our project, we revisit our assumptions and analyze any risk assessment. This in turn would help us improve our program and any design ideas in the future. We assumed that both parties have the correct @uncfsu.edu domain. By providing a non @broncos.uncfsu.edu, the email will not be sent to other parties because of the mailing server causing the other party to never receive the email. We also assume that the requester has the contact information for the other party because the application works primarily on the use of user provided information. Finally, we assumed both parties have access to an active internet connection because this is the only way to access the application and receive emails.

**7. Final Thoughts**

The overall design process of our project had its ups and downs. The team learned how to work with Web2py as well as designing a web based application. The next sections highlight the struggles and what is ahead for our project. The code and documentation can be viewed in the GitHub repository <https://github.com/jangoflyte/Senior-Project>.

**7.1 Struggles**

We had some struggles throughout the application development. For instance, we had to change up the project because our original topic of sexual consent was sensitive and required special paperwork. We had to revise our project, however, we took the overall logic our application and took out the sexual consent out of the picture.

**7.2 What is Ahead**

This was an interesting project because we had sixteen weeks to brainstorm, create, and develop a program to be presented. As mentioned earlier, we had some obstacles within the whole design process but we managed to deliver a working application. If we had more time, we could add SMS messaging within the app. The app only allows a certain mail server so outside mail such as Gmail or Yahoo would not be able to communicate with user. We could in the future add other mail server as well as adding an extra layer of security su as email encryption or having an authentication token. The built in Calendar API was useful to our project, but we could switch to advance APIs such as Google Calendar or iCloud Calendar. Finally, we could modify or improve logic within the application and make the User Interface simpler and nicer. As mentioned earlier, this was a time sensitive project and we wanted to create a working application.

**Bibliography**

1. <http://www.web2py.com/>
2. <https://www.freepascal.org/docs-html/fcl/sqldb/usingsqldb.html>
3. <https://searchexchange.techtarget.com/definition/SMTP>