**Senior Project Final Report**

**Department of Computer Science**

**Calvin College**

Title: Calvin Scouting Report

Author: Ethan Clark and Mitch Stark

Date: May 11, 2017

Mentor: Professor Victor Norman

# Project Vision and Overview

## Background and Problem

The issue is the lack of simple, convenient methods to produce a scouting report on athletic teams in the MIAA Conference. Currently, the Men’s Soccer team at Calvin uses a template word document that is manually filled in to show the starting line-up, top-goal scorers, and other key players. To do this, the coaching staff generally has to go onto the other team’s website, search through games played and see who has started the most, and then continue looking through the stats to determine the “best” players. The Women’s Volleyball team only uses game film to scout opposing teams. This is inefficient and inconvenient to do multiple times in a single week for both teams. Furthermore, because of the manual work required, the coaching staff usually does not have time to dig through deeper and more meaningful statistics, like goals per game average for each player, saves per game for goalies, when teams score most of their goals, etc. for soccer. Finally, many games include small descriptions of all of the goals that have been scored. This could be extremely useful to find out the most common ways that teams score (close-range, off of crosses, etc.) which is simply too much work to do manually.

## Brief Description of Solution Provided

The solution is going to be a simple, convenient, time-saving web application that produces scouting report information about MIAA Athletic Teams. The solution will include a scouting report for the Calvin Men’s Soccer and the Calvin Women’s Volleyball. The solution scrapes data off NCAA Division 3 teams’ websites, including team statistics, individual player statistics, and game logs. This will allow Calvin coaches to get a more in-depth view of a team on a player-by-player basis with a simple click of a button. The user experience is a web application with a tab navigation layout, one tab for soccer, one tab for volleyball, and one tab to add teams.

## Your Interest and Qualifications

Our main interest in this project is because Mitch is on the Calvin Men’s Soccer Team and believes this tool could be useful for both coaches and players at Calvin College. Ethan’s fiancé played for 4 years on the Calvin Women’s Volleyball team and, as a fan for the past four years, wants to give back to the program. Ethan and Mitch are both seniors at Calvin College with programming experience in front-end programming (for the user interface experience) and back-end programming (web scraping, databases).

# Mentor Selection, Expert User and Collaboration

The mentor for this project is Professor Norman. Professor Norman is Mitch’s academic advisor and he frequently attends sporting games at Calvin College. We worked with Coach Ryan Souders (soccer) and Coach Amber Warners (volleyball) specifically about the user interface experience portion of the application and what data/information they want to see in the scouting report. The only dependency we had on outside sources was finding meeting times to meet with the coaches. We tried to meet with the coaches every three weeks.

# Development Approach

The development approach that was used on this project was a combination of a phased approach and an iterative approach. The first phase (fall semester) was developing the web scrapers and the database schema. The second phase (spring semester) was developing the web application that would be the user interface for the consumption of the scouting report data and statistics. Both of these phases had two-week iterations which had deadlines of our goals such as completing plans, prototypes, and features (see the **Development Timeline** below). Finally, we made our development as agile as possible. We had scheduled meetings with our stakeholders, Coach Sounders and Coach Warners, every three weeks to get feedback on the data being collected and the user interface design. As these coaches are the primary users of the solution, the development will be dependent on how what they want to see.

**Development Timeline:**

* October 5: Back-End & Front-End Development Plan Complete
  + Web Scraping Language/Library
  + Database
  + Front-End Language/Library
* October 19: Web Scraper Prototype Complete
* November 2: Database Schema & Populating Database Complete
* November 16: Prototype Web Server - API Development Complete
* November 30: Prototype Web App - Structural Development Complete
* December 5: CSx95 Presentation
* December 14: Prototype Web App - CSS Complete & Project Status Report
* January: Off
* February 15: Web App – Structural Development Complete
* March 1: Web App – Functionality Development Complete
* March 15: Web Server – API Development Complete
  + Web App – CSS Complete
* March 29: User Testing Complete
* April 12: Updates to Application from User Testing
* April 26: Research into Application Extensions
  + Printable PDF
  + Manually refresh stats button to run web scraper
  + More teams for soccer
  + More teams for volleyball
  + Storing team sites in database (instead of hard coding)
* May 4: All Development Complete & Prepare for Presentation
* May 8: Senior Project Presentation
* May 18: Final Deliverables

**System Design**

All code for the project can be found at <https://github.com/EthanMitchSeniorProject>. The following are the repositories that are found in our organization:

* BackEnd
* Database
* WebApp
* Documents

**BackEnd Repo - Web Scraper Scripts**

There are three web scraper files within this repository that make up the data collection BackEnd: web\_scraper.py, game\_web\_scraper.py, vball\_game\_scraper.py.

* + web\_scaper.py: Collects player statistics for both the soccer and volleyball teams that are stored in the database. It uses the following link (which is the same structure for NCAA D3 Schools that use Presto Sports, the URL is just different) to scrape this data:
    - Soccer: <http://calvinknights.com/sports/msoc/2017-18/teams/calvin?view=profile&r=0&pos=kickers>
    - Volleyball: <http://calvinknights.com/sports/wvball/2017-18/teams/calvin?view=profile&r=0&pos>=
  + game\_web\_scraper.py: Collects game statistics/data for soccer teams that are stored in the database. It uses the following link (which is the same structure for NCAA D3 Schools that use Presto Sports, the URL is just different) to scrape this data:
    - <http://calvinknights.com/sports/msoc/2018-19/schedule>
  + vball\_game\_scraper.py: Collects game play-by-play data for volleyball teams that are stored in the database. It uses the following link (which is the same structure for NCAA D3 Schools that use Presto Sports, the URL is just different) to scrape this data:
    - <http://calvinknights.com/sports/wvball/2018-19/schedule>

The scrapers can be run manually using the following commands:

* + python3 web\_scraper.py
  + python3 game\_web\_scraper.py
  + python3 vball\_game\_scraper.py

**Database Repo - Database Implementation**

In the Database repository in GitHub, there are two database schema images (one for soccer and one for volleyball) and a SQL file that implements the database schema. The images are for visual representation of the database schema and how tables are connected. The schema.sql file creates the tables and columns necessary for our database.

Our database is an SQL Database hosted on Microsoft Azure.

* + “SQL Database is a general-purpose relational database managed service in Microsoft Azure that supports structures such as relational data, JSON, spatial, and XML.”

The SQL Database *ScoutingReport* is a single-instance database hosted on the SQL Server *calvinscoutingreport*. The firewall settings on *calvinscoutingreport* are set to let all incoming IP addresses. The server name, that can be used to access the server, is calvinscoutingreport.database.windows.net.

We used VS Code to connect to the SQL Server and Database. Connect to the database through the following steps:

1. Install VS Code.
2. Install *mssql* extension in VS Code.
3. Open the Command Palette in VS Code (can be found under the View menu).
4. Type “MS SQL: Connect” into the Command Palette input menu and click Enter.
5. Click Enter for the “Create Connection Profile” prompt.
6. Enter the following information as prompted:
   1. Server name: calvinscoutingreport.database.windows.net
   2. Database name: ScoutingReport
   3. Authentication Type: SQL Login
   4. Username: athlete
   5. Password: calvinscoutingreport123!
   6. Save Password: yes/no
   7. Profile Name: whatever you want to save the profile, so you don’t have to enter all the information every time

To execute an SQL Script (i.e. run an SQL query against the database), write the query in a file and use the Command Palette to find the “MS SQL: Execute Query” command (keyboard shortcuts vary depending on the OS).

**Web App Repo - Front End Web Application**

Our web application is built using Express.js, which is a web application framework for Node.js, for the server-side code and React JavaScript for the client-side code.

React uses the idea of components to build out the client-side for web applications. Our application has a single index.html file that accesses the Webpack bundled file for ease of use. Our components are broken down into soccer components, volleyball components, and other required components.

The whole application is hosted on a Linux Virtual Machine created for us to use by Chris Wierenga (CS Lab Administrator) that is running in the CS Lab at Calvin College. Here is the information regarding the virtual machine:

* + 2 CPU cores, 8GB of RAM, and 40GB of Hard Drive
  + IP Address: 153.106.116.105
  + DNS Name: 18sp-cs398-scouting.cs.calvin.edu

These are the following steps that must be taken in order to deploy and run the web application on a Linux Virtual Machine:

1. Install git
2. Install npm
3. Install Nodejs
4. Install python3 AND pip3
5. Download our Web App GitHub Repo:
   1. git clone <https://www.github.com/EthanMitchSeniorProject/WebApp>
6. Enter the Web App directory
   1. Retrieve BackEnd submodule using: git submodule update --init --recursive
7. Run: npm install
8. Verify that the Web App works by running:
   1. npm run build
   2. npm run start
9. Install pydobc library AND mssql jdbc 13 driver with pip3
10. Enter the BackEnd directory
    1. Install dependencies as needed, using pip3
11. Verify that web\_scraper.py, game\_web\_scraper.py, and vball\_game\_scraper.py work by running the python3 commands found under **BackEnd Repo - Web Scraper Scripts** section of this document.
    1. The web application must be running for this to work because the scripts use the server.js routes to get the teams to scrape.

The web application can be accessed by going to 18sp-cs398-scouting.cs.calvin.edu.

# Resources

|  |  |  |
| --- | --- | --- |
| **Resource** | **Source/Provider** | **Cash Cost** |
| Linux Virtual Machine | Calvin CS Labs | $0 |
| **Total Cash Cost** | --------------------- | $0 |

## Testing

Our approach for testing was running internal testing for the python scripts and external testing for our web application. We held two user testing sessions each with Coach Souders and Coach Warners for a total of four user testing sessions. We recorded any issues/bugs/recommendations from the sessions and prioritized them based on the amount of work we had.

**Results**

**Deliverables**

Here is a list of the deliverables we completed:

* + Deployed Web Application:
    - <http://18sp-cs398-scouting.cs.calvin.edu>
  + SQL Database
    - Test
  + Web Scraper Scripts
    - <https://github.com/EthanMitchSeniorProject/BackEnd>
  + GitHub
    - <https://github.com/EthanMitchSeniorProject>

**Learning Outcomes**

* + Valuable experience with a project from start to finish
    - This included:
      * Requirements
      * Design
      * Development
      * Testing
      * Presentation
  + Experience with various levels of a system
    - This included:
      * Python (Beautiful Soup Library)
      * SQL
      * JavaScript
        + Node/Express
        + React
  + Practice troubleshooting/debugging errors
  + Front-End Development
    - Our skillset is not in front end web development. Nonetheless, it was good practice for us to design and create the UI for our web app, no matter how it turned out.

**Future Work**

Please find our documented future work below:

**Known Issues:**

* + Only one goalkeeper is displayed if a team has a goalkeeper with number 0 and a goalkeeper with number 00. We store the players’ numbers as integers, but the database/front end sees these numbers as the same so only one is displayed.
    - SOLUTION: Store player numbers as strings and not integers.
  + The soccer modals cannot be opened twice in a row. If a modal is clicked and another modal is clicked, the first modal can be opened again. One just cannot be opened twice in a row.
    - SOLUTION: Utilize the “open” method.
  + TCP Connection error when running the volleyball vball\_game\_scraper.py. The connection seems to time out because this script takes a fairly long time to run.
    - SOLUTION: Unknown

**Incompletes:**

* + Add Team feature does not immediately run the web scrapers.
    - Currently, the team is added to the database and will get run at the next scheduled CRON job. Ideally, the Add Team feature would kick off the web scrapers just for that team.
  + Adding player positions in the soccer view under the starters view.
  + Removing team id and game id from the volleyball drop downs.

**Potential Improvements:**

* + Graphing the soccer player trends in the player modal instead of just displaying the stats.
    - This would provide a visual appeal to the project and can be easier to interpret than just seeing numbers.
  + Graphing the volleyball rotation trends in the volleyball view instead of just displaying the stats.
    - Again, like the bullet above, this would provide a visual appeal to the project and can be easier to interpret than just getting a lot of different numbers on the web page.
  + Secure login for users.
    - Security: only authorized users (coaches from the Calvin soccer team and the Calvin volleyball team could access the site)
  + Display web scraper script errors to user if a particular team fails while scraping data

**Conclusion/Acknowledgments**

We would like to thank Professor Norman for being our trusty advisor for our project this semester. We met with him weekly and provided him exactly what tasks we completed for the previous tasks and exactly what tasks we wanted to complete for the next week. We want to thank him for keeping us accountable and guiding us whenever we had questions.

We would like to thank Chris Wierenga and his help with setting up a Linux Virtual Machine for us to host our web application and python scripts.

We would finally like to thank the two coaches, Coach Ryan Souders and Dr. Amber Warners, for their participation and feedback on our project throughout the year. They are the beneficiaries of the work that we did this semester on the Calvin Scouting Report Project, and we hope our project helps them in their efforts as coaches at Calvin College.

**References**

The following is the list of resources where our work can be found:

* + GitHub
    - <https://github.com/EthanMitchSeniorProject>
  + Microsoft Azure
    - Test
  + Google Doc Notebook
    - <https://docs.google.com/document/d/1o9kIUpcU77SwYokKE7Dy68HFQ1oD6H_XVyUYVOof5ug/edit>

The following is the list of major references that we used throughout the year to learn about libraries and languages for best practices:

* + <https://expressjs.com/en/guide/database-integration.html#mysql>
  + <https://stackoverflow.com/questions/372885/how-do-i-connect-to-a-mysql-database-in-python>
  + <https://docs.microsoft.com/en-us/azure/sql-database/sql-database-connect-query-python>
  + <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>