# Kyle R. Timins

161 Oxford Drive
South Windsor, CT 06074
(860) 212-2254
KyleRTimins@gmail.com

**OBJECTIVE**

To secure a full time position utilizing my over five years of workforce experience along with my own personal learned knowledge.

**EDUCATION**

*Bachelor of Science,* Computer Software Engineering, Vermont Technical College, Randolph Center, VT,
Graduated May 2014

**COMPUTER SKILLS**

*Languages:* Powershell, C#, Powershell, Bash, LaTeX, VB.Net, T-SQL, JSON, XML/XPath, Perl, AutoHotKey
*Operating Systems:* Windows, GNU/Linux, Unix
*Programs:* Vim, Git
*Exposure To:* Ada, Computer security, FreeBSD, CLISP, Java, Scheme, MVC, Javascript

**EXPERIENCE**

*Senior Software Developer* Jun 2014 – March 2020
*Software Engineer && Senior Software Engineer* // Previous Titles
Insurity, Hartford, CT
Multiple Areas

- Worked on Backend of thirteen Property & Causality Lines of Business
  - Maintained (Bug fixes) both Legacy and the new "Evolution" products
  - Created new functionality via both Customer enhancements and Regulatory enhancements
  - Refactored codebase to increase maintainability, readability, and bring them more inline with SOLID principles
  - *Parts of the Backend System:* Rating, Mapping, Statistical Coding, Policy Forms Generation
  - *Lines of Business:* Property Casualty, Business Owners, ISO Inland Marine, AAIS Inland Marine, Ocean Marine, Boiler and Machinery, Commercial Output, Capital Assets, Cargo, and Farm
    - Was part of the original development team to bring up Farm and Cargo
    - Currently learning Liability lines. Such as General Liability and Crime lines of business.
- Worked on interfaces that created the feed to downstream Reporting Systems like WINS and Custom Systems
  - Became the sole Subject Matter Expert (SME) in these areas when the previous SME left less than a month after I started learning these areas
  - Worked with our internal WINS team and Clients' WINS teams to gain a better understanding and knowledge of the WINS system itself
  - Worked with my manager, at that time, on an initiative to distribute knowledge and workload of this to Line of Business teams where applicable. Tasks of this initiative included:
    - Setup training classes/presentations

**EXPERIENCE CONT.**

* Work one on one with those who would be performing harder tasks, such as bringing up new Client or Lines from scratch
* Maintain an "Open Door" policy with helping others
  - Worked on creation of XML that passed information from our Policy system to the Billing system.
    - "Owned" the code for this and am the SME of this code and system
    - Perform enhancement and break-fix work on this system
    - Became a "liaison" between our Policy system team and the Billing system team to increase communication and efficiency, including creating proper procedures on how the two systems and teams work together
    - Completed an effort to bring the XML created into proper compliance with XML namespaces and namespace prefixes
    - Created "mock/fix" XMLs to fix any premium discrepencies between the Policy and Billing systems due to various issue.s
  - Helped create automated systems, behind the scenes, that made labor heavy and mundane tasks into automated processes that went beyond current processes.
    - Created an automated system, in Powershell, that increased reporting and visibility of errors occurring in hosted environments utilizing our current error system that would only create a Zip file of log files and entry in a database. This worked as a "second line of defense" in conjunction with standard environment monitoring systems.
      * Was designed and implemented as a JSON based configured system allowing for a variety of different "run reasons" without having to change the program itself.
      * Allowed the user to pick from a variety of different predetermined categories of errors that were known to happen or define their own set of exception messages or error stacks, utilizing SQL Server wild cards, including the null set to get back all errors. A set of "not" exceptions can also be defined to exclude certain errors, also utilizing SQL Server wild cards, from being reported in the run. Multiple sets could be defined in a config file to give a more encompassing view of the system status. Further fine tuning could be done by giving timespans to look back, a minimum count threshold, per item in the set, the system that threw the error, and the Line(s) of Business associated with the error.
      * The config file allowed for defining a set, one or more, Client and environment pair(s) to define which database to gather information from
      * CSV, JSON, and/or XML files could be created, as defined in the config file, to allow for further reporting of errors.
      * An email could also be created containing the error information, with granular tuning of information included defined by error set. Reporting files could be attached to the email. (Zip files were originally attached as well, but functionally was removed due to email size limitation and replaced with a link to where to get a zip of zips.) A set of email recipients could also be defined based on a defined service pack (aka. sprint) or a specified date range or timespan from a specified start date.
    - Created an automated reporting and notification system of scheduled assignments.
      * The previous process involved creating a spreadsheet of assignments that remained "stagnant" after the start of the service pack (aka. sprint) it was for. After the system was put through a trial run, it was

**EXPERIENCE CONT.**

implemented as official procedure for assignment schedule reporting and notification.
  - Was originally created in Powershell to utilize functionality I created in other scripts. It was later ported over to a full C# application.
  - Was designed and implemented as JSON based configured system allowing for a variety of different "run reasons" without having to modify the program itself. This was used in conjunction with command line arguments to define the tasks desired for that run. Command line arguments also allowed for overwriting the definitions in the config file.
  - Was originally a command line program only. However, a GUI was later added to increase usability of "one off" runs.
  - It utilized a wild card based system to determine what spreadsheet to read in based on a lookup table of conversion between a date span and the defined service pack numbering. Naming could be changed in the config file.
  - It would read in the spreadsheet of assignments and call out to the Assignment Management System via API to get up-to-date information on each assignment including current resource assigned and estimated & completed hours. This could be scheduled via Windows Scheduled Tasks or run manually on-demand. A CSV file, in the Powershell version, or a JSON file, in the C# version, would be created to store information collected so that information reporting could be performed without an update of data.
  - An XLSX file could be created to allow for reporting and analysis. This was later utilized for Assignees to be able to state their current status of their assignments. (i.e. Currently being worked on, delayed, will not make the scheduled due date, etc...)
  - Emails could be created to be sent to the Assignees with their current list, to Managers of all assignments for those under them grouped by Assignee, and Account Managers of all assignments for Clients they are responsible for grouped by Client and Assignee. CCed recipients could be specified, along with a "Do not send to" list in case people didn't want to be included on the emails. As well, a CSV file was utilized defining a lookup of recipients that should be automatically added if a defined recipient was already on the email.
- Spearheaded a project to upgrade to MSXML6 in our legacy system. This utilized our official build system, along with a custom built Powershell build script, and our developer update system that I modified to allow for distributing the DLLs created from a build machine to a test machine that existed outside of our standard build and update system. My role in this was not only to do software upgrades, but also become a "one man" DevOps team in a siloed environment.
- Various, random tasks that covered all parts of the company.

*IT Summer Intern* Jun 2013 – Aug 2013
Travelers Ins., Hartford, CT
OAI-SYS Department – Information - Systems Security

- Created Perl 5 scripts to perform metrics on compliance
- Created and maintained database of Portable Media Exceptions
- Created basic list of endpoint security
- Capstone on current and potential use of mobile devices in Travelers' Claim Department

**EXPERIENCE CONT.**

*IT Summer Intern* Jun 2012 – Aug 2012
Travelers Ins., Hartford, CT
OAI-SYS Department – Service Management

- Created Perl 5 scripts to increase efficiency and automation of Server Audits
- Performed Server Audit
- Headed XMatters add-on research to increase escalation efficiency
- Worked on N05/C03 patches for BMC TM ART monitoring system
- Capstone on current and potential use of automation in Travelers

*IT Summer Intern* May 2011 – Aug 2011
Travelers Ins., Hartford, CT
OAI-SYS Department – Advanced Technologies

- Created Perl 5 scripts to increase efficiency and automation
- Fixed web applications to allow for cross-browser compatibility
  - Allowed custom tool tip pop-up to be used in non Internet Explorer browsers
  - Create copy from browser work-around for non Internet Explorer browsers
- Remotely reinstalled firewall device operating system and created written documentation
- Organized retired Cisco networking equipment for recycling
- Capstone on potential use of open source software in Travelers