CMSC 447

Software Development Plan (SDP)

[1 Scope 6](#_gjdgxs)

[1.1 Identification 6](#_30j0zll)

[1.2 System overview 6](#_1fob9te)

[1.3 Document overview 6](#_3znysh7)

[1.4 Relationship to other plans 6](#_tyjcwt)

[2 Referenced documents 6](#_3dy6vkm)

[3 Overview of required work 6](#_4d34og8)

[4 Plans for performing general software development activities 7](#_17dp8vu)

[4.1 Software development process 7](#_26in1rg)

[4.2 General plans for software development 7](#_1ksv4uv)

[4.2.1 Software development methods 7](#_44sinio)

[4.2.2 Standards for software products 7](#_z337ya)

[4.2.3 Reusable software products 8](#_4i7ojhp)

[4.2.4 Handling of critical requirements 8](#_3whwml4)

[4.2.5 Computer hardware resource utilization 8](#_1pxezwc)

[4.2.6 Recording rationale 8](#_49x2ik5)

[4.2.7 Access for acquirer review 8](#_2p2csry)

[5 Plans for performing detailed software development activities 9](#_3o7alnk)

[5.1 Project planning and oversight 9](#_ihv636)

[5.1.1 Software development planning (covering updates to this plan) 9](#_32hioqz)

[5.1.2 CSCI test planning 9](#_1hmsyys)

[5.1.3 System test planning 9](#_41mghml)

[5.1.4 Software installation planning 9](#_2grqrue)

[5.1.5 Software transition planning 9](#_vx1227)

[5.1.6 Following and updating plans, including the intervals for management review 9](#_3fwokq0)

[5.2 Establishing a software development environment 9](#_4f1mdlm)

[5.2.1 Software engineering environment 9](#_2u6wntf)

[5.2.2 Software test environment 9](#_19c6y18)

[5.2.3 Software development library 9](#_3tbugp1)

[5.2.4 Software development files 9](#_28h4qwu)

[5.2.5 Non-deliverable software 9](#_nmf14n)

[5.3 System requirements analysis 9](#_1mrcu09)

[5.3.1 Analysis of user input 10](#_46r0co2)

[5.3.2 Operational concept 10](#_2lwamvv)

[5.3.3 System requirements 10](#_111kx3o)

[5.4 System design 10](#_206ipza)

[5.4.1 System-wide design decisions 10](#_4k668n3)

[5.4.2 System architectural design 10](#_2zbgiuw)

[5.5 Software requirements analysis 10](#_3ygebqi)

[5.6 Software design 10](#_sqyw64)

[5.6.1 CSCI-wide design decisions 10](#_1rvwp1q)

[5.6.2 CSCI architectural design 10](#_4bvk7pj)

[5.6.3 CSCI detailed design 10](#_2r0uhxc)

[5.7 Software implementation and unit testing 10](#_1664s55)

[5.7.1 Software implementation 10](#_3q5sasy)

[5.7.2 Preparing for unit testing 10](#_25b2l0r)

[5.7.3 Performing unit testing 10](#_kgcv8k)

[5.7.4 Revision and retesting 10](#_34g0dwd)

[5.7.5 Analyzing and recording unit test results 10](#_1jlao46)

[5.8 Unit integration and testing 11](#_2iq8gzs)

[5.8.1 Preparing for unit integration and testing 11](#_xvir7l)

[5.8.2 Performing unit integration and testing 11](#_3hv69ve)

[5.8.3 Revision and retesting 11](#_1x0gk37)

[5.8.4 Analyzing and recording unit integration and test results 11](#_4h042r0)

[5.9 CSCI qualification testing 11](#_1baon6m)

[5.9.1 Independence in CSCI qualification testing 11](#_3vac5uf)

[5.9.2 Testing on the target computer system 11](#_2afmg28)

[5.9.3 Preparing for CSCI qualification testing 11](#_pkwqa1)

[5.9.4 Dry run of CSCI qualification testing 11](#_39kk8xu)

[5.9.5 Performing CSCI qualification testing 11](#_1opuj5n)

[5.9.6 Revision and retesting 11](#_48pi1tg)

[5.9.7 Analyzing and recording CSCI qualification test results 11](#_2nusc19)

[5.10 CSCI/HWCI integration and testing 11](#_3mzq4wv)

[5.10.1 Preparing for CSCI/HWCI integration and testing 11](#_2250f4o)

[5.10.2 Performing CSCI/HWCI integration and testing 11](#_haapch)

[5.10.3 Revision and retesting 11](#_319y80a)

[5.10.4 Analyzing and recording CSCI/HWCI integration and test results 11](#_1gf8i83)

[5.11 System qualification testing 11](#_2fk6b3p)

[5.11.1 Independence in system qualification testing 12](#_upglbi)

[5.11.2 Testing on the target computer system 12](#_3ep43zb)

[5.11.3 Preparing for system qualification testing 12](#_1tuee74)

[5.11.4 Dry run of system qualification testing 12](#_4du1wux)

[5.11.5 Performing system qualification testing 12](#_2szc72q)

[5.11.6 Revision and retesting 12](#_184mhaj)

[5.11.7 Analyzing and recording system qualification test results 12](#_3s49zyc)

[5.12 Preparing for software use 12](#_meukdy)

[5.12.1 Preparing the executable software 12](#_36ei31r)

[5.12.2 Preparing version descriptions for user sites 12](#_1ljsd9k)

[5.12.3 Preparing user manuals 12](#_45jfvxd)

[5.12.4 Installation at user sites 12](#_2koq656)

[5.13 Preparing for software transition 12](#_3jtnz0s)

[5.13.1 Preparing the executable software 12](#_1yyy98l)

[5.13.2 Preparing source files 12](#_4iylrwe)

[5.13.3 Preparing version descriptions for the support site 12](#_2y3w247)

[5.13.4 Preparing the "as built" CSCI design and other software support information 12](#_1d96cc0)

[5.13.5 Updating the system design description 12](#_3x8tuzt)

[5.13.6 Preparing support manuals 12](#_2ce457m)

[5.13.7 Transition to the designated support site 12](#_rjefff)

[5.14 Software configuration management 13](#_1qoc8b1)

[5.14.1 Configuration identification 13](#_4anzqyu)

[5.14.2 Configuration control 13](#_2pta16n)

[5.14.3 Configuration status accounting 13](#_14ykbeg)

[5.14.4 Configuration audits 13](#_3oy7u29)

[5.14.5 Packaging, storage, handling, and delivery 13](#_243i4a2)

[5.15 Software product evaluation 13](#_338fx5o)

[5.15.1 In-process and final software product evaluations 13](#_42ddq1a)

[5.15.2 Software product evaluation records, including items to be recorded 13](#_2hio093)

[5.15.3 Independence in software product evaluation 13](#_wnyagw)

[5.16 Software quality assurance 13](#_3gnlt4p)

[5.16.1 Software quality assurance evaluations 13](#_1vsw3ci)

[5.16.2 Software quality assurance records, including items to be recorded 13](#_4fsjm0b)

[5.16.3 Independence in software quality assurance 13](#_2uxtw84)

[5.17 Corrective action 13](#_3u2rp3q)

[5.17.1 Problem/change reports, including items to be recorded (candidate items include project name, originator, problem number, problem name, software element or document affected, origination date, category and priority, description, analyst assigned to the problem, date assigned, date completed, analysis time, recom- mended solution, impacts, problem status, approval of solution, follow-up actions, corrector, correction date, version where corrected, correction time, description of solution implemented) 14](#_2981zbj)

[5.17.2 Corrective action system 14](#_odc9jc)

[5.18 Joint technical and management reviews 14](#_1nia2ey)

[5.18.1 Joint technical reviews, including a proposed set of reviews 14](#_47hxl2r)

[5.18.2 Joint management reviews, including a proposed set of reviews 14](#_2mn7vak)

[5.19 Other software development activities 14](#_3ls5o66)

[5.19.1 Risk management, including known risks and corresponding strategies 14](#_20xfydz)

[5.19.2 Software management indicators, including indicators to be used 14](#_4kx3h1s)

[5.19.3 Security and privacy 14](#_302dr9l)

[5.19.4 Subcontractor management 14](#_1f7o1he)

[5.19.5 Interface with software independent verification and validation (IV&V) agents 14](#_3z7bk57)

[5.19.6 Coordination with associate developers 14](#_2eclud0)

[5.19.7 Improvement of project processes 14](#_thw4kt)

[5.19.8 Other activities not covered elsewhere in the plan 14](#_3dhjn8m)

[6 Schedules and activity network 14](#_4cmhg48)

[7 Project organization and resources 15](#_3qwpj7n)

[7.1 Project organization 15](#_261ztfg)

[7.2 Project resources 15](#_356xmb2)

[8 Notes 15](#_44bvf6o)

[9 A. Appendixes 15](#_3im3ia3)

# Scope

This section shall be divided into the following paragraphs.

## Identification

This paragraph shall contain a full identification of the system and the software to which this document applies, including, as applicable, identification number(s), title(s), abbreviation(s), version number(s), and release number(s).

## System overview

This system aims to provide information about areas to move to within the United States consistent with specified criteria. Each specified criteria will appear as a search option on a webpage and will be used by a user to narrow down results for a new location to move to. Locations fitting the criteria of the search will be displayed on a map as points.

## Document overview

This paragraph shall summarize the purpose and contents of this document and shall describe any security or privacy considerations associated with its use.

## Relationship to other plans

This paragraph shall describe the relationship, if any, of the SDP to other project management plans.

# Referenced documents

This section shall list the number, title, revision, and date of all documents referenced in this plan. This section shall also identify the source for all documents not available through normal Government stocking activities.

# Overview of required work

The project will be divided into two major sections, the front end and the back end. The front end will consist of a web page with parameters able to be set by the user and relayed to the back end. The back end will consist of a server holding information relating to conditions relevant to the search. The information will be pulled from various government agencies to ensure quality. The information will be sent to the database and relevant searches returned to the front end. From there, the relevant information will be relayed to a Google Maps API, which will drop pins on the map at the chosen locations. When the pins are hovered over by the user, relevant information will be shown to further inform the user.

First, possible sources for the database will be researched to decide what information will best fit the parameters. Second, design of the font and back ends will begin in parallel to meet the timeline and ensure compatibility between them. Thirdly, the parts will be integrated together and tested. Finally, the project will be tested, all documentation proof read and finalized, and the project presented and turned over to the customer.

# Plans for performing general software development activities

## Software development process

The project will be developed using the agile method, with weekly and biweekly sprints to accomplish goals. A Gantt chart breaking down the sprints is available below in section 6. At the end of the sprint the group will meet in person on Thursdays at 2:30 PM with Professor Marron to confirm the work completed was satisfactory, and prepare for the next sprint.

## General plans for software development

### Software development methods

list all tools?

This paragraph shall describe or reference the software development methods to be used. Included shall be descriptions of the manual and automated tools and procedures to be used in support of these methods. The methods shall cover all contractual clauses concerning this topic. Reference may be made to other paragraphs in this plan if the methods are better described in context with the activities to which they will be applied.

### Standards for software products

This paragraph shall describe or reference the standards to be followed for representing requirements, design, code, test cases, test procedures, and test results. The standards shall cover all contractual clauses concerning this topic. Reference may be made to other paragraphs in this plan if the standards are better described in context with the activities to which they will be applied. Standards for code shall be provided for each programming language to be used. They shall include at a minimum:

* + - 1. Standards for format (such as indentation, spacing, capitalization, and order of information)
      2. Standards for header comments (requiring, for example, name/identifier of the code; version identification; modification history; purpose; requirements and design decisions implemented; notes on the processing (such as algorithms used, assumptions, constraints, limitations, and side effects); and notes on the data (inputs, outputs, variables, data structures, etc.)
      3. Standards for other comments (such as required number and content expectations)
      4. Naming conventions for variables, parameters, packages, procedures, files, etc.
      5. Restrictions, if any, on the use of programming language constructs or features

* + - 1. Restrictions, if any, on the complexity of code aggregates

### Reusable software products

#### Incorporating reusable software products

This paragraph shall describe the approach to be followed for identifying, evaluating, and incorporating reusable software products, including the scope of the search for such products and the criteria to be used for their evaluation. It shall cover all contractual clauses concerning this topic. Candidate or selected reusable software products known at the time this plan is prepared or updated shall be identified and described, together with benefits, drawbacks, and restrictions, as applicable, associated with their use.

DataUSA Api, Google Docs, AWS (if we use it)

#### Developing reusable software products

This paragraph shall describe the approach to be followed for identifying, evaluating, and reporting opportunities for developing reusable software products. It shall cover all contractual clauses concerning this topic.

Scope detector

### Handling of critical requirements

This paragraph shall be divided into the following subparagraphs to describe the approach to be followed for handling requirements designated critical. The planning in each subparagraph shall cover all contractual clauses concerning the identified topic.

#### Safety assurance

#### Security assurance

#### Privacy assurance

#### Assurance of other critical requirements

### Recording rationale -- Note taking process

At every meeting one of the members present will take minutes and record all comments, concerns, and confirmations set forth by Professor Marron. From these notes an email confirmation reiterating what was discussed will be set to all members and Professor Marron. A complete log of all emails will also be stored on the github with the rest of the documentation when the project is completed.

In addition, any discussions after class or outside of regular meetings will be sent as an email to all members of the group to distribute work as needed, confirm smaller changes, and provide assistance to other group members in between sprint presentations.

This paragraph shall describe the approach to be followed for recording rationale that will be useful to the support agency for key decisions made on the project. It shall interpret the term "key decisions" for the project and state where the rationale are to be recorded. It shall cover all contractual clauses concerning this topic.

### Access for acquirer review -- Maybe retool for access to documents with GitHub

This paragraph shall describe the approach to be followed for providing the acquirer or its authorized representative access to developer and subcontractor facilities for review of software products and activities. It shall cover all contractual clauses concerning this topic.

# Plans for performing detailed software development activities

This section shall be divided into the following paragraphs. Provisions corresponding to non-required activities may be satisfied by the words "Not applicable." If different builds or different software on the project require different planning, these differences shall be noted in the paragraphs. The discussion of each activity shall include the approach (methods/procedures/tools) to be applied to: 1) the analysis or other technical tasks involved, 2) the recording of results, and 3) the preparation of associated deliverables, if applicable. The discussion shall also identify applicable risks/uncertainties and plans for dealing with them. Reference may be made to [4.2.1](#_1xrdshw) if applicable methods are described there.

## Project planning and oversight

### Software development planning (covering updates to this plan) -- How we plan software development?

Planning will take place after classes on Tuesday and Thursday nights and following bi-weekly meetings with our customer, Dr. Marron on Thursday afternoons. In addition, weekend and evening work sessions and further planning will be scheduled as needed. Planning will be confirmed by an email sent by Kyle Coleman sent after the meetings to confirm work schedules.

### CSCI test planning -- Components

Individual components will be tested as they are created, and as they are integrated into the system. In addition, user tests with volunteers providing feedback on website usability and aesthetic. The customer, Dr. Marron, will have final say on the appearance and layout of the front end of the project.

### System test planning -- Entire System

As the CSCI’s are integrated into the system periodic tests will be run to ensure stability. Following these stress tests to ensure that the API’s and databases can be repeatedly called without error will be run to ensure that they return correct results to the front end in a prompt manner.

### Software installation planning -- Access Website, no install

The project is a website that will be hosted on a local computer for the purposes of simplicity of this project. In addition, a local SQL server will be used in conjunction with API’s to keep the time and cost of data retrieval to a minimum.

### Software transition planning -- None

### Following and updating plans, including the intervals for management review -- Keep empty until something changes

## Establishing a software development environment

This paragraph shall be divided into the following subparagraphs to describe the approach to be followed for establishing, controlling, and maintaining a software development environment. The planning in each subparagraph shall cover all contractual clauses regarding the identified topic.

### Software engineering environment -- Google Docs and GitHub

Google Docs will be the writing environment for the documentation of the project, using the templates provided by Russ Cain. Project code will be stored and developed on GitHub to ensure quality control and group consensus on design decisions. Code will be developed on local machines and then pushed to GitHub, followed by pulls from group members to ensure compatibility across multiple devices.

### Software test environment -- Manual Testing

Testing will be done at multiple stages, as both the CSCI’s are created and as they are integrated into the system. An overall test for functionality will be done when the system is functional and ready to be presented for approval to the customer. Testing at each level will be done by all members of the group to ensure consensus and to navigate any issues that arise.

### Software development library -- Standard PHP, HTML, CSS, Google Maps API, SQL

The front end of the project will be a webpage in HTML with CSS wrappers for appearance, supported by a PHP framework for accessing the database / API’s. In addition, the Google Maps API will be used for generating a map to visualize the locations generated by the project as prospective candidates. The database holding a copy of the information used by the search parameters will be coded in SQL.

### Software development files -- None

### Non-deliverable software -- GitHub

Software that is not part of the final project, such as proof of concept wireframes, will be cataloged as it is built upon through backups on GitHub. These older editions will remain there as reference points for later work on the project and as a backlog of stable older editions.

## System requirements analysis

This paragraph shall be divided into the following subparagraphs to describe the approach to be followed for participating in system requirements analysis. The planning in each subparagraph shall cover all contractual clauses regarding the identified topic.

### Analysis of user input --

Requirements will be confirmed with the customer, Dr. Marron before they are integrated into the project. User inputs will be based on the search parameters defined by Dr. Marron. The parameters will be selected by means of buttons and sliding bars so as to tailor the search for the user. Front end requirements will be added to the project as specified, and the databases and website frameworks will be tailored to the back end requirements. Should any questions arise, they will be clarified with Dr. Marron before moving forward.

### Operational concept --

User needs will be mapped to requirements during the Software Design Description (SDD) and Software Requirements Specification (SRS) sections to follow. In addition, user tests from volunteers will be used to provide feedback on how the website can be streamlined, with final input and approval coming from Dr. Marron.

### System requirements --

To fulfill the requirements proposed by Dr. Marron, the project will consist of both a webpage front end and a database / API backend. The frontend will require an internet browser and internet access to function properly, while the backend will require internet access for the API traffic and a local copy of an SQL database. The project will be hosted locally on the user’s machine, avoiding the need for calls to an external server.

## System design

This paragraph shall be divided into the following subparagraphs to describe the approach to be followed for participating in system design. The planning in each subparagraph shall cover all contractual clauses regarding the identified topic.

### System-wide design decisions -- Made as a team

The site will consist of a single web page containing a map and a form for the user to specify their search query. After a search is performed by the user, the results will show up on the map as pins. If a user clicks on a pin then a popup will show up with details about the location.

### System architectural design -- Made as a team

We plan to use the Google Maps Javascript API in addition to bootstrap for the front end portion of the site. For the backend we will use Apache and PHP. Ideally we be able to get all of our data from APIs as we need it, but if that is too slow or if we have to use CSVs as data sources, then we will create an SQL database. When a user goes to the webpage and submits the form specifying their search criteria, the javascript will pass the request to the backend, which will query an API and/or the database and return an array of location results. The javascript code in the frontend will then interact with the Google Maps API to plot the resulting locations on the map.

## Software requirements analysis

After receiving the requirements we will research and find data sources such as APIs and CSV files that contain the data needed to be able to satisfy the requirements regarding the search criteria. Then we will work to narrow down the data sources into the smallest set that still provides all the data needed to satisfy the requirements. If a particular search criteria requirement is unfeasible given the data available, then we will discuss this with our client and work to adjust the requirement appropriately.

## Software design

*This paragraph shall be divided into the following subparagraphs to describe the approach to be followed for software design. The planning in each subparagraph shall cover all contractual clauses regarding the identified topic.*

### CSCI-wide design decisions -- Made as a team

The frontend will use the Google Maps Javascript API, JQUERY, and Bootstrap. The backend will be PHP (using an Apache server).

### CSCI architectural design -- Made as a team

*Frontend webpage*: This webpage will contain an embedded google maps for displaying results and a search form for specifying desired search parameters.

*search.php*: This backend file will receive search parameters from the frontend and then return an array of locations that resulted from the search.

*Database*: If we decide we need a database, we will create two tables:

* states: This table will contain the name (and a unique identifier) for each state.
* counties: This table will contain the name (and a unique identifier) for each county and the ID of the corresponding State that it belongs to. It will also contain a column for each data parameter that we decide to store about the county.

### CSCI detailed design -- Made as a team

*Frontend webpage*: The frontend of the site will consist of a static HTML file, CSS file, and JS file. The user will be able to click a “hamburger” button to have a form appear with search criteria options. The form will contain a search button at the bottom that will hide the form and cause the results to appear on the map.When the search form is submitted, the Javascript will store the form’s values as JSON and then send a POST request to search.php using the JQUERY library. When results are returned the Javascript will use the Google Maps API to display a pin on the map for each search result.

*search.php*: This backend file will receive parameters from a POST request made by the frontend. This file will then parse the parameters and use them to query the API and/or the database and then return the results to the POST request as a JSON array of locations (such as county names or coordinates).

## Software implementation and unit testing

This paragraph shall be divided into the following subparagraphs to describe the approach to be followed for software implementation and unit testing. The planning in each subparagraph shall cover all contractual clauses regarding the identified topic.

### Software implementation -- Divided up by skill and time constraint

Code like no one is watching! Make mistakes and learn from them, write your mistakes down and track your progress. I don’t know how to implement software… sorry

### Preparing for unit testing -- Unit tests developed in respect to current code

### Performing unit testing -- Tests performed in respect to current code

### Revision and retesting -- Regressions performed at end of each week sprint

### Analyzing and recording unit test results -- Tests success will be recorded at end of each week sprint

## Unit integration and testing

This paragraph shall be divided into the following sub- paragraphs to describe the approach to be followed for unit integration and testing. The planning in each subparagraph shall cover all contractual clauses regarding the identified topic.

### Preparing for unit integration and testing

### Performing unit integration and testing

### Revision and retesting

### Analyzing and recording unit integration and test results

## CSCI qualification testing

This paragraph shall be divided into the following sub- paragraphs to describe the approach to be followed for CSCI qualification testing. The planning in each subparagraph shall cover all contractual clauses regarding the identified topic.

### Independence in CSCI qualification testing -- No independence because no outside testers

### Testing on the target computer system -- There is no one target computer system, SAFARI is the only requirement, may remove

### Preparing for CSCI qualification testing -- Have mock results to compare against actual result

### Dry run of CSCI qualification testing -- Actual comparisons of those results

### Performing CSCI qualification testing -- Actual Actual comparison of those results

### Revision and retesting -- Change code when errors appear

### Analyzing and recording CSCI qualification test results -- Revisions recorded by Github

## System qualification testing

This paragraph shall be divided into the following sub- paragraphs to describe the approach to be followed for participating in system qualification testing. The planning in each subparagraph shall cover all contractual clauses regarding the identified topic.

FOLLOW PREVIOUS SECTION BUT REFERENCE ENTIRE SYSTEM

### Independence in system qualification testing

### Testing on the target computer system

### Preparing for system qualification testing

### Dry run of system qualification testing

### Performing system qualification testing

### Revision and retesting

### Analyzing and recording system qualification test results

## Preparing for software use

Code Fury will release this software as open source with instructions for downloading from our github account to set up on a server for personal or business use.

### Preparing version descriptions for user sites

Code Fury will prepare the software for delivery to Dr. Marron. This preparation will be subdivided into batch files, command files, data files or other software files needed to install and operate the software on a server, we will also provide instructions for implementation to include command line scripts for setup.

### Preparing user manuals

A user manual will consist of the executable instructions we will outline as a README on our github account.

## Software configuration management

## The SCM will define and establish uniform steps to control and maintain our product. This will be a best effort as directed by our task leader in managing and maintain future changes. The following items will refer to source code, documents and diagrams.

### Configuration identification

This will be a stand alone product, but we will establish a 4 digit identifier assigned by the task leader to each product. A revision number will be assigned to follow the 4 digit identifier which will follow as xxxx.1 and upon reaching xxxx.1.9 another incrementation as follows xxxx.2.1

### Configuration control

After a stable version of our software is established individual software will then be lumped into one version which we may title SR-1(Software Release). Revisions will follow the scheme identified above, SR-1.1, SR-1.2.. etc

### Configuration status accounting

Status of our progress will be established and recorded during office hour visits with Dr. Marron and team meetings as directed by the task leader.

### Configuration audits

Bi-weekly internal audits may be conducted by wither our task leader or another suitable person who wishes to track our progress.

### Packaging, storage, handling, and delivery

All maintenance will be conducted in our github repository which is open for any who wish to view our software development.

## Software product evaluation

How well we meet our development requirements will directly influence our evaluation. A grade will be assigned by Dr. Marron and Prof. Cain upon completion of our assignment.

### In-process and final software product evaluations

May 1st, 2018 is our target date for final evaluation. May 10th, 2018 is our expected software demonstration day to be conducted in the Bldg. Fine Arts room 303 at about 5:30 pm.

### Software product evaluation records, including items to be recorded

Each team member will provide commits with detailed comments as to what they are pushing, a log of these changes can be recorded and tracked to show work performed.

## Software quality assurance

Task leader will establish testing protocols which will utilize black box methods. Tests will search for possible errors and that we have satisfied our system requirements.

Process audits will be conducted to find deficiencies and help close them out. Task leader will assign SQA audits of the software against the SDP.

Task leader will resolve the discrepancies with SQA and software manager.

### Software quality assurance evaluations

Unit tests will be conducted as a group, walkthroughs will be conducted during our weekly team meetings so that all eyes can assessed possible issues with codes and any conflicts will be addressed as a team.

### Software quality assurance records, including items to be recorded

Github will have annotations of bugs we have found and resolutions that address those issues.

## Corrective action

This paragraph shall be divided into the following subparagraphs to describe the approach to be followed for corrective action. The planning in each subparagraph shall cover all contractual clauses regarding the identified topic.

### Problem/change reports, including items to be recorded (candidate items include project name, originator, problem number, problem name, software element or document affected, origination date, category and priority, description, analyst assigned to the problem, date assigned, date completed, recommended solution, impacts, problem status, approval of solution, follow-up actions, corrector, correction date, version where corrected, correction time, description of solution implemented) -- Github Issues can probably fulfill all these things

### Corrective action system --Github Issues, fix your stuff if it has issues

## Joint technical and management reviews

This paragraph shall be divided into the following subparagraphs to describe the approach to be followed for joint technical and management reviews. The planning in each subparagraph shall cover all contractual clauses regarding the identified topic.

### Joint technical reviews, including a proposed set of reviews -- Explain design reviews with each other

### Joint management reviews, including a proposed set of reviews -- Cain tells us when we’re late

## Other software development activities

This paragraph shall be divided into the following subparagraphs to describe the approach to be followed for other software development activities. The planning in each subparagraph shall cover all contractual clauses regarding the identified topic.

### Risk management, including known risks and corresponding strategies -- Risk Matrix

### Software management indicators, including indicators to be used -- I don’t know what this is

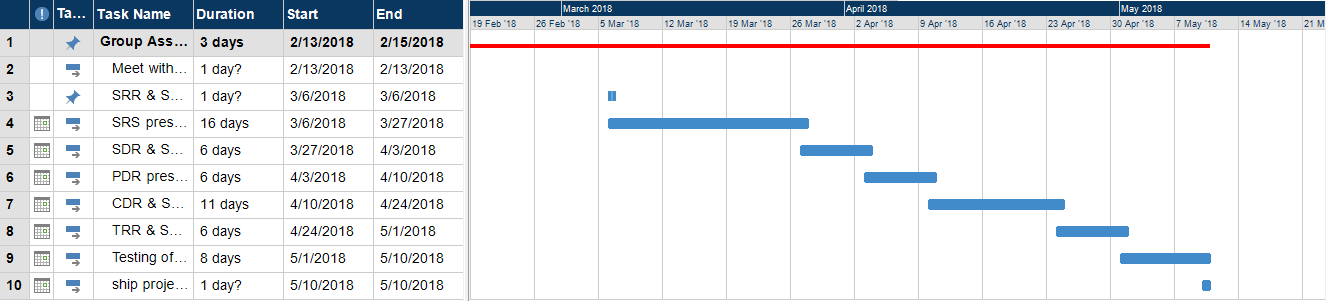
### Security and privacy -- Not relevant to us

### Other activities not covered elsewhere in the plan -- Use this if necessary

# Schedules and activity network

The current schedule is to have bi-weekly meetings with Dr. Marron to discuss progress on the project, approve work, and receive feedback and clarifications.

* + Feb 13 - Group assigned and project proposed
  + Feb 15 - Meet with Dr. Marron to discuss requirements, begin work on SRR & SDP
  + Mar 6 - SRR & SDP presented to Dr. Marron for approval, begin work on SDR & SRS
  + Mar 13 - SRS presented to Dr. Marron for approval, begin work on SDD. Begin gathering API data and research options
  + Mar 27 - SDR & SDD presented to Dr. Marron for approval, begin work on PDR. Begin work on project front end and back end
  + Apr 3 - PDR presented to Dr. Marron for approval, begin work on CDR & STD
  + Apr 10 - CDR & STP presented to Dr. Marron for approval, begin work on TRR & STR
  + Apr 24 - TRR & STR presented to Dr. Marron for approval, begin testing project, begin work on SAR & SUM
  + May 1 - Testing of project complete, project shippable. present project, SAR, and SUM to Dr. Marron for approval. Make final corrections as needed
  + May 10 - ship project. Present project in class for peer and teacher review. Turn over all materials to both Dr. Marron and Russ Cain.



# Project organization and resources

The organizational structure will consist of a task leader, design engineer, developers, customer liaison and technical writers. At anytime additional duties can be established by the task leader to help distribute the work flow amongst the group. The task leader reports directly to Prof. Cain, issues that require resolution to involve the customer will be a joint effort with customer liaison and the task leader.

## Project organization

Task Leader - Kyle Coleman

Design Engineers: (front end) Brian Sapp, (back end) Dan Engbert

Technical Writer: Tyler Karlsen

Quality Assurance: Rushmie Kulkarni

Customer Liaison: Jon Danko

Developers: Doug Gessleman, Jon Danko, Tyler Karlsen, Rushmie Kulkarni

Other tasks as assigned by task leader.

## Project resources

The software development will be supported by the following items:

1. Personnel resources, including:
   1. 7 personnel
   2. Virtual Machines to implement servers, Microsoft Word, Microsoft Excel
   3. Dan Engbert: PHP, SQL, Python, C++, HTML, CSS, Javascript; Kyle Coleman: (insert skills); Brian Sapp: (insert skills); Tyler Karlsen: (insert skills), Rushmie Kulkarni: (insert skills); Jon Danko: C, C++, Java, Python, PHP, SQL; Doug Gessleman: C, C++, Java, Python, Visual Basic .NET
2. Github as a repository, UMBC AOK Library for meetings, classroom for presentation.
3. Individual personal laptops acquired by each individual.

# Notes

This section shall contain any general information that aids in understanding this document (e.g., background information, glossary, rationale). This section shall include an alphabetical listing of all acronyms, abbreviations, and their meanings as used in this document and a list of any terms and definitions needed to understand this document.

# Appendixes

Appendixes may be used to provide information published separately for convenience in document maintenance (e.g., charts, classified data). As applicable, each appendix shall be referenced in the main body of the document where the data would normally have been provided. Appendixes may be bound as separate documents for ease in handling. Appendixes shall be lettered alphabetically (A, B, etc.).

DESCRIPTION/PURPOSE

The Software Development Plan (SDP) describes a developer’s plans for conducting a software development effort. The term "software development" is meant to include new development, modification, reuse, reengineering, maintenance, and all other activities resulting in software products.

The SDP provides the acquirer insight into, and a tool for monitoring, the processes to be followed for software development, the methods to be used, the approach to be followed for each activity, and project schedules, organization, and resources.

APPLICATION/INTERRELATIONSHIP

Portions of this plan may be bound separately if this approach enhances their usability. Examples include plans for software configuration management and software quality assurance.

The Contract Data Requirements List (CDRL) should specify whether deliverable data are to be delivered on paper or electronic media; are to be in a given electronic form (such as ASCII, CALS, or compatible with a specified word processor or other support software); may be delivered in developer format rather than in the format specified herein; and may reside in a computer-aided software engineering (CASE) or other automated tool rather than in the form of a traditional document.

PREPARATION INSTRUCTIONS

General instructions.

a. Automated techniques. Use of automated techniques is encouraged. The term "document" in this means a collection of data regardless of its medium.

b. Alternate presentation styles. Diagrams, tables, matrices, and other presentation styles are acceptable substitutes for text when data required can be made more readable using these styles.

c. Title page or identifier. The document shall include a title page containing, as applicable: document number; volume number; version/revision indicator; security markings or other restrictions on the handling of the document; date; document title; name, abbreviation, and any other identifier for the system, subsystem, or item to which the document applies; contract number; CDRL item number; organization for which the document has been prepared; name and address of the preparing organization; and distribution statement. For data in a database or other alternative form, this information shall be included on external and internal labels or by equivalent identification methods.

d. Table of contents. The document shall contain a table of contents providing the number, title, and page number of each titled paragraph, figure, table, and appendix. For data in a database or other alternative form, this information shall consist of an internal or external table of contents containing pointers to, or instructions for accessing, each paragraph, figure, table, and appendix or their equivalents.

e. Page numbering/labeling. Each page shall contain a unique page number and display the document number, including version, volume, and date, as applicable. For data in a database or other alternative form, files, screens, or other entities shall be assigned names or numbers in such a way that desired data can be indexed and accessed.

f. Response to tailoring instructions. If a paragraph is tailored out of this document, the resulting document shall contain the corresponding paragraph number and title, followed by "This paragraph has been tailored out." For data in a database or other alternative form, this representation need occur only in the table of contents or equivalent.

g. Multiple paragraphs and subparagraphs. Any section, paragraph, or subparagraph in this DID may be written as multiple paragraphs or subparagraphs to enhance readability.

h. Standard data descriptions. If a data description required by this document has been published in a standard data element dictionary specified in the contract, reference to an entry in that dictionary is preferred over including the description itself.

i. Substitution of existing documents. Commercial or other existing documents, including other project plans, may be substituted for all or part of the document if they contain the required data.