CMSC 447

Software Design Description (SDD)

[1 Scope 3](#_Toc432621786)

[1.1 Identification 3](#_Toc432621787)

[1.2 System overview 3](#_Toc432621788)

[1.3 Document overview 3](#_Toc432621789)

[2 Referenced documents 3](#_Toc432621790)

[3 CSCI-wide design decisions 3](#_Toc432621791)

[4 CSCI architectural design 4](#_Toc432621792)

[4.1 CSCI components 4](#_Toc432621793)

[4.2 Concept of execution 5](#_Toc432621794)

[4.3 Interface design 5](#_Toc432621795)

[4.3.1 Interface identification and diagrams 6](#_Toc432621796)

[4.3.2 (Project-unique identifier of interface) 6](#_Toc432621797)

[5 CSCI detailed design 7](#_Toc432621798)

[5.1 (Project-unique identifier of a software unit, or designator of a group of software units) 8](#_Toc432621799)

[6 Requirements traceability 9](#_Toc432621800)

[7 Notes 9](#_Toc432621801)

[A. Appendixes 10](#_Toc432621802)

# 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 paragraph shall briefly state the purpose of the system and the software to which this document applies. It shall describe the general nature of the system and software; summarize the history of system development, operation, and maintenance; identify the project sponsor, acquirer, user, developer, and support agencies; identify current and planned operating sites; and list other relevant documents.

The goal of this project is to create a web application which will allow a user to create an account, and from such account be able to evaluate different statistical values such that a list of locations within the United States will be outputted to the user which correlate to the user input. With the list, a map should be presented with pins that correlate to the location and can then be further expanded to output a location image or description.

We expect to use a database for maintaining username and password data. Along with this we expect to be using third party software for our output. This would include the use of GoogleMaps for our map output, and possibly Flickr for our image outputs, and Wikipedia for a description output. Further third-party software will be expanded in this section as seems fit throughout the project.

Because the nature of this project is in the scope of a classroom project, there will not be long term maintenance of the project and will be run on a local machine. Throughout this document the term sponsor will refer to John Winder who is the group’s client throughout the project. Similarly, the group refers to the group of developers working on the project including Matthew Hearn, Aaron Lewis, Alex Rochford, Cathy Poore, Ben Kittner, and Steven Heckman. The project will refer to the software and documentation created for this assignment.

All project development will be done through GitHub and then developer preference for development environments, debuggers, ect. The web portion of the application will be implemented through UMBC’s student web environment(SWE.) Due to this, as per UMBC’s DoIT guidelines, all user data inputted into a webpage is stored in an insecure manner which will be further noted within the web application. The reference for further information on the SWE is found on <https://wiki.umbc.edu/pages/viewpage.action?pageId=67994741>.

## 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.

The goal of this document is to serve as a description of the system design as well as how the system will be split into different components. This document will all list any frameworks, APIs, or any other component which will need to be integrated into the system.

# Referenced documents

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

SDD – Software Design Document

SRS- Software Requirement Specification

# CSCI-wide design decisions

This section shall be divided into paragraphs as needed to present CSCI-wide design decisions, that is, decisions about the CSCI’s behavioral design (how it will behave, from a user’s point of view, in meeting its requirements, ignoring internal implementation) and other decisions affecting the selection and design of the software units that make up the CSCI. If all such decisions are explicit in the CSCI requirements or are deferred to the design of the CSCI’s software units, this section shall so state. Design decisions that respond to requirements designated critical, such as those for safety, security, or privacy, shall be placed in separate subparagraphs. If a design decision depends upon system states or modes, this dependency shall be indicated. Design conventions needed to understand the design shall be presented or referenced. Examples of CSCI-wide design decisions are the following:

1. Design decisions regarding inputs the CSCI will accept and outputs it will produce, including interfaces with other systems, HWCIs, CSCIs, and users [(4.3.x](#_bookmark3) of this DID identifies topics to be considered in this description). If part or all of this information is given in Interface Design Descriptions (IDDs), they may be referenced.
2. Design decisions on CSCI behavior in response to each input or condition, including actions the CSCI will perform, response times and other performance characteristics, description of physical systems modeled, selected equations/algorithms/rules, and handling of unallowed inputs or conditions.
3. Design decisions on how databases/data files will appear to the user [(4.3.x](#_bookmark3) of this DID identifies topics to be considered in this description). If part or all of this information is given in Database Design Descriptions (DBDDs), they may be referenced.
4. Selected approach to meeting safety, security, and privacy requirements.
5. Other CSCI-wide design decisions made in response to requirements, such as selected approach to providing required flexibility, availability, and maintainability.

We do not currently have any CSCI wide design decisions. All design decisions are made at the individual CSCI unit level.

# CSCI architectural design

This section shall be divided into the following paragraphs to describe the CSCI architectural design. If part or all of the design depends upon system states or modes, this dependency shall be indicated. If design information falls into more than one paragraph, it may be presented once and referenced from the other paragraphs. Design conventions needed to understand the design shall be presented or referenced.

## CSCI components

This paragraph shall:

1. Identify the software units that make up the CSCI. Each software unit shall be assigned a project-unique identifier.

Note: A software unit is an element in the design of a CSCI; for example, a major subdivision of a CSCI, a component of that subdivision, a class, object, module, function, routine, or database. Software units may occur at different levels of a hierarchy and may consist of other software units. Software units in the design may or may not have a one- to-one relationship with the code and data entities (routines, procedures, databases, data files, etc.) that implement them or with the computer files containing those entities. A database may be treated as a CSCI or as a software unit. The SDD may refer to software units by any name(s) consistent with the design methodology being used.

The overall system we are creating is a web application which will include a login, account modification, a questionnaire, and map output. Each of these are the components of the system, which upon completion will be a different version of the design. All these also work with one another, however can be tested on an individual basis.

CSCI component 1 will be the login portion of the application. This will include the original homepage of the website which will allow a user to create an account or login. Upon successful login the user would access the homepage for the application. For the login, the application will need to interface with the login database which will contain a user index, username, password, and “About Me” textfield. Creating account should increment the index, as well as verify that the username portion is unique, and neither the username or password are NULL.

CSCI component 2 will be the account modification portion of the application. This will be available from the homepage and allow the user to change their username, password, or “About Me” textfield. Modifying the account will also need to interface with the login database. Modifying the account should verify that the new username is unique, neither the username or password are NULL, as well as the username and password are different from the previously used username and password.

CSCI component 3 will be the questionnaire portion of the application. This will need to present the user different statistics and allow the user to give a ranking to each statistic based on personal preference. The component will then need to interface with a dataset which holds statistical data to output locations which best correspond to the user preferences. \*\*\*asked Ben in GroupMe how were are interfacing with the data\*\*\*

CSCI component 3 will be the a map output which will visualize the data collected in the questionnaire. This will use a GoogleMaps API to printout a map of the United States, and then place pins on the top locations for the user. From there, a user can click on the pins and using a Flickr API, a picture of the location will be displayed, and using a Wikipedia API a brief description of the location will be displayed below the picture. \*\*\*verify with everyone else that they found APIs for these components \*\*\*

1. Show the static (such as "consists of") relationship(s) of the software units. Multiple relationships may be presented, depending on the selected software design methodology (for example, in an object-oriented design, this paragraph may present the class and object structures as well as the module and process architectures of the CSCI).
2. State the purpose of each software unit and identify the CSCI requirements and CSCI- wide design decisions allocated to it. (Alternatively, the allocation of requirements may be provided in [6.a.)](#_bookmark5)
3. Identify each software unit’s development status/type (such as new development, existing design or software to be reused as is, existing design or software to be reengineered, software to be developed for reuse, software planned for Build N, etc.) For existing design or software, the description shall provide identifying information, such as name, version, documentation references, library, etc.
4. Describe the CSCI’s (and as applicable, each software unit’s) planned utilization of computer hardware resources (such as processor capacity, memory capacity, input/output device capacity, auxiliary storage capacity, and communications/network equipment capacity). The description shall cover all computer hardware resources included in resource utilization requirements for the CSCI, in system-level resource allocations affecting the CSCI, and in resource utilization measurement planning in the Software Development Plan (SDP). If all utilization data for a given computer hardware resource are presented in a single location, such as in one SDD, this paragraph may reference that source. Included for each computer hardware resource shall be:
   1. The CSCI requirements or system-level resource allocations being satisfied
   2. The assumptions and conditions on which the utilization data are based (for example, typical usage, worst-case usage, assumption of certain events)
   3. Any special considerations affecting the utilization (such as use of virtual memory, overlays, or multiprocessors or the impacts of operating system overhead, library software, or other implementation overhead)
   4. The units of measure used (such as percentage of processor capacity, cycles per second, bytes of memory, kilobytes per second)
   5. The level(s) at which the estimates or measures will be made (such as software unit, CSCI, or executable program)
5. Identify the program library in which the software that implements each software unit is to be placed.

## Concept of execution

This paragraph shall describe the concept of execution among the software units. It shall include diagrams and descriptions showing the dynamic relationship of the software units, that is, how they will interact during CSCI operation, including, as applicable, flow of execution control, data flow, dynamically controlled sequencing, state transition diagrams, timing diagrams, priorities among units, handling of interrupts, timing/sequencing relationships, exception handling, concurrent execution, dynamic allocation/deallocation, dynamic creation/deletion of objects, processes, tasks, and other aspects of dynamic behavior.

## Interface design

This paragraph shall be divided into the following subparagraphs to describe the interface characteristics of the software units. It shall include both interfaces among the software units and their interfaces with external entities such as systems, configuration items, and users. If part or all of this information is contained in Interface Design Descriptions (IDDs), in section [5](#_bookmark4) of the SDD, or elsewhere, these sources may be referenced.

### Interface identification and diagrams

This paragraph shall state the project-unique identifier assigned to each interface and shall identify the interfacing entities (software units, systems, configuration items, users, etc.) by name, number, version, and documentation references, as applicable. The identification shall state which entities have fixed interface characteristics (and therefore impose interface requirements on interfacing entities) and which are being developed or modified (thus having interface requirements imposed on them). One or more interface diagrams shall be provided, as appropriate, to depict the interfaces.

### (Project-unique identifier of interface)

This paragraph (beginning with 4.3.2) shall identify an interface by project-unique identifier, shall briefly identify the interfacing entities, and shall be divided into subparagraphs as needed to describe the interface characteristics of one or both of the interfacing entities. If a given interfacing entity is not covered by this SDD (for example, an external system) but its interface characteristics need to be mentioned to describe interfacing entities that are, these characteristics shall be stated as assumptions or as "When [the entity not covered] does this, [the entity that is covered] will ...." This paragraph may reference other documents (such as data dictionaries, standards for protocols, and standards for user interfaces) in place of stating the information here. The design description shall include the following, as applicable, presented in any order suited to the information to be provided, and shall note any differences in these characteristics from the point of view of the interfacing entities (such as different expectations about the size, frequency, or other characteristics of data elements):

* + - 1. Priority assigned to the interface by the interfacing entity(ies)
      2. Type of interface (such as real-time data transfer, storage-and-retrieval of data, etc.) to be implemented
      3. Characteristics of individual data elements that the interfacing entity(ies) will provide, store, send, access, receive, etc., such as:
         1. Names/identifiers

Project-unique identifier

Non-technical (natural-language) name

DoD standard data element name

Technical name (e.g., variable or field name in code or database)

Abbreviation or synonymous names

* + - * 1. Data type (alphanumeric, integer, etc.)
        2. Size and format (such as length and punctuation of a character string)
        3. Units of measurement (such as meters, dollars, nanoseconds)
        4. Range or enumeration of possible values (such as 0-99)
        5. Accuracy (how correct) and precision (number of significant digits)
        6. Priority, timing, frequency, volume, sequencing, and other constraints, such as whether the data element may be updated and whether business rules apply
        7. Security and privacy constraints
        8. Sources (setting/sending entities) and recipients (using/receiving entities)
      1. Characteristics of data element assemblies (records, messages, files, arrays, displays, reports, etc.) that the interfacing entity(ies) will provide, store, send, access, receive, etc., such as:
         1. Names/identifiers

Project-unique identifier

Non-technical (natural language) name

Technical name (e.g., record or data structure name in code or database)

Abbreviations or synonymous names

* + - * 1. Data elements in the assembly and their structure (number, order, grouping)
        2. Medium (such as disk) and structure of data elements/assemblies on the medium
        3. Visual and auditory characteristics of displays and other outputs (such as colors, layouts, fonts, icons and other display elements, beeps, lights)
        4. Relationships among assemblies, such as sorting/access characteristics
        5. Priority, timing, frequency, volume, sequencing, and other constraints, such as whether the assembly may be updated and whether business rules apply
        6. Security and privacy constraints
        7. Sources (setting/sending entities) and recipients (using/receiving entities)
      1. Characteristics of communication methods that the interfacing entity(ies) will use for the interface, such as:
         1. Project-unique identifier(s)
         2. Communication links/bands/frequencies/media and their characteristics
         3. Message formatting
         4. Flow control (such as sequence numbering and buffer allocation)
         5. Data transfer rate, whether periodic/aperiodic, and interval between transfers
         6. Routing, addressing, and naming conventions
         7. Transmission services, including priority and grade
         8. Safety/security/privacy considerations, such as encryption, user authentication, compartmentalization, and auditing
      2. Characteristics of protocols the interfacing entity(ies) will use for the interface, such as:
         1. Project-unique identifier(s)
         2. Priority/layer of the protocol
         3. Packeting, including fragmentation and reassembly, routing, and addressing
         4. Legality checks, error control, and recovery procedures
         5. Synchronization, including connection establishment, maintenance, termination
         6. Status, identification, and any other reporting features
      3. Other characteristics, such as physical compatibility of the interfacing entity(ies) (dimensions, tolerances, loads, voltages, plug compatibility, etc.)

# CSCI detailed design

This section shall be divided into the following paragraphs to describe each software unit of the CSCI. If part of all of the design depends upon system states or modes, this dependency shall be indicated. If design information falls into more than one paragraph, it may be presented once and referenced from the other paragraphs. Design conventions needed to understand the design shall be presented or referenced. Interface characteristics of software units may be described here, in Section [4,](#_bookmark0) or in Interface Design Descriptions (IDDs). Software units that are databases, or that are used to access or manipulate databases, may be described here or in Database Design Descriptions (DBDDs).

## (Project-unique identifier of a software unit, or designator of a group of software units)

This paragraph shall identify a software unit by project-unique identifier and shall describe the unit. The description shall include the following information, as applicable. Alternatively, this paragraph may designate a group of software units and identify and describe the software units in subparagraphs. Software units that contain other software units may reference the descriptions of those units rather than repeating information.

* + 1. Unit design decisions, if any, such as algorithms to be used, if not previously selected
    2. Any constraints, limitations, or unusual features in the design of the software unit
    3. The programming language to be used and rationale for its use if other than the specified CSCI language
    4. If the software unit consists of or contains procedural commands (such as menu selections in a database management system (DBMS) for defining forms and reports, on- line DBMS queries for database access and manipulation, input to a graphical user interface (GUI) builder for automated code generation, commands to the operating system, or shell scripts), a list of the procedural commands and reference to user manuals or other documents that explain them
    5. If the software unit contains, receives, or outputs data, a description of its inputs, outputs, and other data elements and data element assemblies, as applicable. Paragraph [4.3.x](#_bookmark3) of this DID provides a list of topics to be covered, as applicable. Data local to the software unit shall be described separately from data input to or output from the software unit. If the software unit is a database, a corresponding Database Design Description (DBDD) shall be referenced; interface characteristics may be provided here or by referencing section [4](#_bookmark0) or the corresponding Interface Design Descriptions (IDDs).
    6. If the software unit contains logic, the logic to be used by the software unit, including, as applicable:
       1. Conditions in effect within the software unit when its execution is initiated
       2. Conditions under which control is passed to other software units
       3. Response and response time to each input, including data conversion, renaming, and data transfer operations
       4. Sequence of operations and dynamically controlled sequencing during the software unit’s operation, including:
          1. The method for sequence control
          2. The logic and input conditions of that method, such as timing variations, priority assignments
          3. Data transfer in and out of memory
          4. The sensing of discrete input signals, and timing relationships between interrupt operations within the software unit
       5. Exception and error handling

# Requirements traceability

This section shall contain:

1. Traceability from each software unit identified in this SDD to the CSCI requirements allocated to it. (Alternatively, this traceability may be provided in [4.1.)](#_bookmark1)
2. Traceability from each CSCI requirement to the software units to which it is allocated.

# 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 Design Description (SDD) describes the design of a Computer Software Configuration Item (CSCI). It describes the CSCI-wide design decisions, the CSCI architectural design, and the detailed design needed to implement the software. The SDD may be supplemented by Interface Design Descriptions (IDDs) and Database Design Descriptions (DBDDs).

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.