Mission: The UMBC Covid-19 Tracker - provide Covid-19 related case and vaccine information for public safety and knowledge

DID: Software Development Plan (SDP)

PREPARED by: Quinque Equos

(

[Alex Wilson](mailto:alexanw1@umbc.edu)

Safia Shah

[Israel Morocho](mailto:cx59298@umbc.edu)

[Duncan Taylor](mailto:ak58697@umbc.edu)

[Tom Tennant](mailto:tent1@umbc.edu)

[David Kravets](mailto:kravets1@umbc.edu)

)

DATE: FEB-2021

REVISION LOG

| Revision | By | Description | Date |
| --- | --- | --- | --- |
| A | Team | Initial description of the SDP | 02/11/21 |
|  |  |  |  |

**Table of Contents**

1. [**Scope.**](#_th3vc4yu5fzd) **5**

[1.1 Identification](#_pizoeg6yqdaj) 5

[1.2 System Overview](#_blt6cfhlybmq) 5

[1.3 Document Overview](#_2zzueskq05ht) 5

[1.4 Relationship with other Plans](#_vc4v7zvgy18d) 5

1. [**Referenced Documents**](#_l8wyaf3hl1le) **5**
2. [**Overview of Required Work**](#_dwzfi76c3fvs) **5**

[3a. Requirements and constraints on the system and software to be developed](#_5ozaxiatkan7) 6

[3b. Requirements and constraints on project documentation](#_m0kpjj16eiyc) 6

[3c. Position of the project in the system life cycle](#_klhk7h41t10p) 6

[3d. The selected program/acquisition strategy or any requirements or constraints on it](#_jt4ayszeqcln) 6

[3e. Requirements and constraints on project schedules and resources](#_23xmj4on1oz8) 6

[3f. Other requirements and constraints, such as on project security, privacy, methods, standards, interdependencies in hardware and software development, etc.](#_mi7rq4vyz5py) 6

[**4. Plans for performing general software development activities.**](#_po45d2dg8l49) **6**

[4.1 Software development process.](#_jlau9w78smhd) 6

[4.2 General plans for software development.](#_27hb0kb6206p) 7

[4.2.1 Software development methods.](#_1sw9xwbh7cxk) 7

[4.2.2 Standards for software products.](#_4blfrbqx3ca9) 7

[4.2.3 Reusable software products.](#_fzrtmstwicyo) 7

[4.2.3.1 Incorporating reusable software products.](#_8qp7wbj3k1ur) 7

[4.2.3.2 Developing reusable software products.](#_o71zxe2q0hf0) 8

[4.2.4 Handling of critical requirements.](#_bnkjo9bgoy85) 8

[4.2.4.1 Safety assurance](#_w590fnsgqen9) 8

[4.2.4.2 Security assurance](#_h3aevzsfvdvy) 8

[4.2.4.3 Privacy assurance](#_daje9i1yo9o8) 8

[4.2.5 Computer hardware resource utilization.](#_nmav845h5ny9) 8

[4.2.6 Recording rationale.](#_c70re3c6egwm) 8

[4.2.7 Access for acquirer review.](#_2l1k774gxoak) 8

[**5. Plans for performing detailed software development activities.**](#_7vm0ykw25dq) **9**

[5.1 Project planning and oversight.](#_hbdfctawxuu8) 9

[5.2 Establishing a software development environment.](#_uujsfl9gwp1m) 9

[5.3 System requirements analysis.](#_q578d6cbi92w) 10

[5.4 System design.](#_p23skpi2fia6) 10

[5.5 Software requirements analysis.](#_jcn7wm4mgrgc) 10

[5.6 Software design.](#_h7k87cc79vtc) 10

[5.7 Software implementation and unit testing.](#_cy1qm597xua9) 11

[5.8 Unit integration and testing.](#_qe1uafiz8prt) 11

[5.9 CSCI qualification testing.](#_7unqwy61hoiz) 11

[CSCI qualification testing is not relevant to this application as it is fully encapsulated in itself and is not dependent on external software](#_9cfrjawkvtma) 11

[5.10 CSCI/HWCI integration and testing.](#_x3gb8vdy3f3q) 12

[CSCI integration and testing is not relevant to this application as it is fully encapsulated in itself and is not dependent on external software](#_mpxf5koign2) 12

[5.11 System qualification testing.](#_qdisd1tec72a) 12

[5.12 Preparing for software use.](#_xaakqgp4033x) 12

[5.13 Preparing for software transition.](#_3p65utabww9v) 12

[5.14 Software configuration management.](#_97ncg82huv22) 12

[5.15 Software product evaluation.](#_khgv0nhioe5p) 12

[5.16 Software quality assurance.](#_aehmd62pmtxf) 13

[5.17 Corrective action.](#_wpkbohgktdc1) 13

[5.18 Joint technical and management reviews.](#_lv139fdyq2sl) 13

[5.19 Other software development activities.](#_hlg12buuoatd) 13

[**6. Schedules and network activity work.**](#_dqspd4txm2h0) **14**

[**7. Project organization and resources.**](#_4ia98qlqc8a9) **14**

[7.1 Project organization.](#_223v2s2ceeg) 15

[7.2 Project Resources](#_kzoeiob4c61d) 16

# Scope.

## **1.1 Identification**

This Software Development Plan pertains to the development of The UMBC Covid-19 Tracker version 1.0. The intended audience includes the engineers tasked with developing the application and the test conductors responsible for the verification and acceptance of the application.

## **1.2 System Overview**

The UMBC Covid-19 Tracker web application is an application meant to take in data from Broadstreet Covid-19 Data Project and display data corresponding to certain states and the counties within them for public knowledge and safety. The web application will also provide the convenience of vaccine data and information. The user can watch and confirm covid cases and mortality cases by county in an available state.

## 

## 1.3 Document Overview

This SDP will give an outline and overview what the web application “name” will do. Section 1 describes the scope of this software. Section 2 provides referenced documents. Section 3 presents an overview of the required work. Plans for general and detailed software development activities are included in Sections 4 and 5, respectively. A schedule and the activity network can be found in Section 6. The project organization is included in Section 7. Privacy and security concerns are considered beyond the scope of this project.

## 1.4 Relationship with other Plans

This paragraph shall describe the relationship, if any, of the SDP to other project management plans. In this instance, there is no relationship between this case and other project management plans.

# Referenced Documents

*// include vaccine information links/ research*

*// include any databases references or resources used to make program*

# Overview of Required Work

This section shall be divided into paragraphs as needed to establish the context for the planning described in later sections.

## 3a. Requirements and constraints on the system and software to be developed

* 1. Take note of what software will be used. Will be found in sections 4 (software design) and 5 (system design)

## 3b. Requirements and constraints on project documentation

* 1. Include a README file for the front end, backend, and application as a whole.
  2. README comments will be included for all classes, API, and major elements.

## 3c. Position of the project in the system life cycle

* 1. The project’s topic is the only element confirmed

## 3d. The selected program/acquisition strategy or any requirements or constraints on it

*Not applicable for this project*

## 3e. Requirements and constraints on project schedules and resources

* 1. Processed data must be up to date. There will be weekly status updates regarding progress on the application.

## 3f. Other requirements and constraints, such as on project security, privacy, methods, standards, interdependencies in hardware and software development, etc.

# 4. Plans for performing general software development activities.

## 4.1 Software development process.

TheUMBC Covid-19 Tracker will be developed using an incremental sectionalized development approach. We will be developing this software in parts which can be linked together at a later stage in the development. The pieces will have minimal dependencies on each other. The pieces of the software will be sectioned out into different tasks between members in a way where they can almost function independently. Everything, in the end, will be connected, but for development purposes, they will need minimal interaction between each other.

This process allows for members to work independently of each other in order to decrease time conflicts. When it comes to linking things together, there will have to be group meetings between multiple people (not always all) in order to coordinate how the pieces will interact. Some pieces will be developed mostly independently but others will be a coordinated effort between multiple members.

## 4.2 General plans for software development.

### 4.2.1 Software development methods.

The information in this section is subject to change as final decisions have not been made. Code will be developed using commercially available products and free programming languages. Developers can use the development environment of their choosing and do not have to match the other developers. This is a web application, so there is no need to differentiate systems.

For version and code tracking, we will be using GitHub. GitHub has built in version control and allows multiple developers to work on the same code without interfering with each other. This service also serves as backups in the event that something occurs with an individual’s computer. We will be working out of a testing branch which when confirmed to work properly will be merged with the master repository.

We will be using Google Drive / Docs in order to work on the data items. This allows for all developers to work on the same document and there is no need to merge them later. Everyone can see what each other is working on and can change / suggest changes on sections at any point. This solution also works as the backup and can be edited by any of our developers from any device with no need to transfer documents when changing work stations.

Google drive: <https://drive.google.com/drive/u/1/folders/0AIYWzm-brlndUk9PVA>

GitHub Repository: <https://github.com/dak003/CS447_covid_tracker>

### 4.2.2 Standards for software products.

We will be somewhat following IEEE standards which have been roughly taught throughout earlier UMBC courses. Some areas may deviate between developers but expectations will be the same for everyone. Code must be commented in a way that any of the other developers could look at it and roughly follow what is happening. Indentation, capitalization, variable naming conventions, ect will be decided upon by members of the subteams and may or may not be documented. Pushes to Git should be commented properly so everyone knows what was added and for what purpose on every push

### 4.2.3 Reusable software products.

UMBC Covid-19 Tracker may have pieces which are reusable between other sections of the software but may or may not be useful outside of the scope of this software.

#### 4.2.3.1 Incorporating reusable software products.

We will be using an API which allows for data to be visualized across maps. This will most likely be heatmap.js or something similar for visualization. We will also be using a front end to back end DB linking software in order to store and interact with data. Queries will be made sure to be parameterized for safety concerns. End users should not be able to change queries for the main part of the software but maybe for side parts.

#### 4.2.3.2 Developing reusable software products.

The idea of being able to turn data into maps is not a new idea but this may be useful for other projects in which the idea is near the same.

### 4.2.4 Handling of critical requirements.

#### 4.2.4.1 Safety assurance

Safety is not a concern within a web application. This will be almost, if not all, informational. There should be little to no chance of someone injured from this web application.

#### 4.2.4.2 Security assurance

Security (along with privacy) is among the top of our concerns. Our web application will ensure that users' computers will not be vulnerable to viruses or anything of the sort from our domain. As mentioned in 4.2.3.1, we will be using parameterized queries in order to protect against simple SQL injection attacks that could leave our database vulnerable. We will be reviewing the code of other developers on our team to ensure this is done properly.

#### 4.2.4.3 Privacy assurance

User privacy is the other in our top concerns. All users should feel safe to use this site in order to gain information and better protect themselves from Covid-19. If it is decided that we will allow users to input any information into the site, we will be using a secure username and password system which will ensure user data and information is protected.

### 4.2.5 Computer hardware resource utilization.

UMBC Covid-19 Tracker is a web application and as such, it requires internet access but does not take up any space on the user’s computer. This page will in no way be intensive of processing power or load times and will be as lightweight as possible for a web application.

### 4.2.6 Recording rationale.

Decisions regarding changes to functionality and design of the application will be made over Discord or Blackboard Collaborate in either voice or text chats. Anything that needs to be confirmed by either the professor or customer will be written and stored in the Google Drive linked in 4.2.1. This will be by form of mainly text documents but is not limited to such.

### 4.2.7 Access for acquirer review.

This section will not be relevant for our software as it will not be shared widely.

# 5. Plans for performing detailed software development activities.

## 5.1 Project planning and oversight.

The software development team will be staffed by a team of 6 members assigned to 3 different subsets of the project: Database configuration and management, REST API development, and front-end UI development. Each sub-team will be further broken down into shared roles of both Software Developer and Acceptance Tester. Collaboration during programming of source code will be necessary to create appropriate test plans for new features either frontend or backend and the subsequent creation of those test cases. Each team will utilize a peer review system to check each other’s work and sufficiently test the code in ways the author might not have attempted (This includes attempting to use the feature in an unintended way, providing incorrect inputs, or searching for uncaught exceptions / edge cases. Testing will be done incrementally per feature added, rather than all at once at the end, and will be a critical part of the peer review process.

## 5.2 Establishing a software development environment.

The software development environment will vary from developer to developer, and could potentially be running operating systems such as Windows 10, MacOSX, or various distributions of Linux. Luckily, because of the nature of platform ambiguity in, for example, python and javascript development, the OS platform will be a non-issue in terms of development dependencies.

In terms of listed integrated development environments (IDE), one has not been made mandatory to use, but suggestions for IDE use with builtin git support include:

* Visual Studio Code
* Atom
* Pycharm

Other editors that do not include native git support that could also be used include:

* Notepad++
* Or other builtin text editing software such as notepad for windows or vi or nano for Linux

The unit tests will be a part of the non-deliverable software, and will be stored in the same repository as the source code. All states of failed tests will be captured in logs and placed in a history directory to keep track of previously failed tests. This directory will also be a part of the github repository.

## 5.3 System requirements analysis.

User input for this web app will mostly be confined to basic front end interaction, and as such, will need very little backend validation after test cases have been written to verify the functionality. All user input will be managed and verified on the front-end to simplify backend endpoints as much as possible.

An operational requirement includes managing the requested dataset and it’s matching visual representation on the front-end. Once a dataset has been requested from the API, the front-end must update according to the user input. As such, the current modules loaded will be kept track of before and after the request is made to avoid mis-displaying the tracker data for both covid cases and vaccines.

System requirements for the user will just be an internet accessible device with a modern web browser. Hosting machines will need the ability to handle simultaneous requests from multiple users, and to maintain performance metrics as will be defined by the customer.

## 5.4 System design.

System design will be managed by the respective teams for small details (teams being broken up by database management, front-end, and back-end) and will come together as a larger group for more general decisions to keep the process of development smooth and streamlined for those not directly involved. Architectural decisions that will affect members of another development team will be brought up in these group discussions as well.

## 5.5 Software requirements analysis.

The software requirements as described in 5.3 will first have tests written, and then manually verified by the developers. Later on, those requirements will be verified by other members of the team, not a part of the development process to additionally verify. Requirements brought up by the customer will be tracked and implemented with a similar process as listed above, spearheaded by that functions development team, but will finally be verified by the customer to ensure all requirements are met.

## 

## 5.6 Software design.

The web app will utilize a REST api to handle backend requests from a javascript front-end. The requests will be a result of direct user interaction of elements on the page, to filter datasets, and manipulate different formats of that data. The current design will not be required to verify string input from the user, but instead just button clicks on embedded elements. The flow will go, front-end requests to REST API, API endpoints to query database tables and fetch appropriate data corresponding to user action.

## 5.7 Software implementation and unit testing.

The software implementation will be done on a feature basis for all requirements. Each developer will handle writing both the source code and tests for their changes in a separate feature branch. All tests must be in a working / pass state before the feature is added to the rest of the project.All front-end changes will require both a visual and functional check by the developer to ensure working order before passing on to the rest of the team. Backend changes will require more express unit testing, and should be developed with the tests at the same time, or the tests beforehand. Python unit tests will be used to verify functionality. Once this process is complete verification of the functionality of the feature must be manually verified and passed on to the rest of the team. If for any reason, the feature fails verification either by the unit test or manual check by team members, changes must be made to the source code and subsequently the tests to match. Unit test results will easily be managed with a pass / fail status given at the end of each test. Those results will be stored in a log file after each iteration in a project wide directory to maintain historical data.

## 5.8 Unit integration and testing.

Selenium Integration tests will be used to validate the front end component of the application, and Python integration tests used to validate larger clusters of backend modules. These tests will help with the manual process of verification of front-end changes. As such, they will be used in tandem with manual verification and will test front-end and monitor the back-end responses from the API. These tests will be run for both front-end and back-end changes to ensure functionality of the application as a whole isn’t negatively affected. Similarly to unit testing, these results will be given a pass/fail status and stored in log files in a separate directory maintaining the historical logs and iterations.

## 5.9 CSCI qualification testing.

### CSCI qualification testing is not relevant to this application as it is fully encapsulated in itself and is not dependent on external software

## 5.10 CSCI/HWCI integration and testing.

### CSCI integration and testing is not relevant to this application as it is fully encapsulated in itself and is not dependent on external software

## 5.11 System qualification testing.

System qualification testing will not be an area of concern for the given application. Modern web browsers handle the idiosyncrasies found between different operating systems.

## 5.12 Preparing for software use.

There will be no executable nor installation required to use the application other than a web browser which is provided in many operating systems.

## 5.13 Preparing for software transition.

All updates to the web application will be handled incrementally to changes on both the front-end and back-end, without any required update to the user. All dependencies both javascript libraries and python libraries will be updated on the server hosting the web application.

## 5.14 Software configuration management.

Source code for the UMBC Covid-19 Tracker will be maintained on GitHub in a git repository. Git will manage version control and was chosen because of its effectiveness in allowing multiple developers to work on the same project at the same time. A secondary feature that will be heavily useful is the repo branch functionality that will allow developers to create development branches for their features, that can be merged into the master branch after being reviewed by the team. Commits made by developers must be labeled with a unique identifier associated with the feature or bug it addresses. The repo can be found here [<https://github.com/dak003/CS447_covid_tracker>**]** and is publicly viewable by anyone. The documents associated with the project will be stored on google drive in a private google drive folder shared to the development team here [<https://drive.google.com/drive/u/2/folders/0AIYWzm-brlndUk9PVA>]. This drive will allow all team members access to the most up to date version of the files while also being accessible by everyone at the same time.

## 5.15 Software product evaluation.

Product evaluation will be done in 2 sections. First during development the product will be internally reviewed by the team during check-in meetings to ensure requirements are being met and progress being made. Final product evaluation will be done with the customer to ensure the functionality and expectations are met. Final product evaluation will NOT be the only time the customer checks-in for progress, but instead will be a final precursor before the product is moved to final stages and out of active production. Records of the evaluation during development will be recorded in the google drive so they are in an easy to format and accessible place for the team to share with the customer if need be.

## 5.16 Software quality assurance.

Software quality will be checked by the development team both in incremental feature update testing at the unit and integration level, as well as during group meetings addressing larger changes to the project as a whole. Regular incremental checks for quality assurance will be vital in maintaining an on target schedule for production.

## 5.17 Corrective action.

Corrective action will entail creating separate git branches based on reported unit and integration test failures as well as changes as a result of product evaluation (**5.15**) and failure of quality assurance (**5.16**). These branches will be labeled as fixes, and will be placed under similar scrutiny as additional feature development for both automated and manual verification.

## 5.18 Joint technical and management reviews.

Joint reviews will take place regularly throughout development and will be mandatory for the merging of any feature branch into the master of the project. These pull requests will initially be verified by the developers on the subset of the project, and later verified by at least one other member of the team before approval. Upon approval commits will be merged into master and the feature or bug branch will be destroyed.

## 5.19 Other software development activities.

There is no corresponding risk with the project nor it’s development. The user is not required to provide any discerning information about themselves and as such privacy is not a concern either. A project schedule has been formulated and will be further elaborated on later in the cycle of the project, as will changes to project processes to deal with any existing inefficiencies.

# 6. Schedules and network activity work.

The timeline for this semester is listed on the table below:

| Feb - 09 - 2021 | Feb - 25 - 2021 | Mar - 11 - 2021 | Apr - 08 - 2021 | May - 06 - 2021 |
| --- | --- | --- | --- | --- |
| Phase 0 | Phase 1 | Phase 2 | Phase 3 | Phase 4 |
| Start up | Data gathering and environment development | Implementation | Assembly | Testing and Maintenance |
| Project Idea is formed  Roles are Assigned | Collect the necessary data that will display the number of Covid Cases and begin plans on obtaining an environment that will be able to hold said data that we can later build upon | Transitioning Covid data into a format that can be visualized across a map as well as creating a user interface | Joining both the front and back end coding along with any other pieces that were assembled individually to produce a final product | The creation of testing scripts to correct any underlying issues as well as including any possible quality of life improvements. |

# 

# 7. Project organization and resources.

The project is structured into 5 different phases that are listed below:

**Phase 0:** Our startup phase, where our team roles are assigned and a general overview of our project plan is defined

**Phase 1:** Obtaining the necessary data for our application. Thankfully there are many database sources to pick from, what's left is to choose which database will be most useful to our product. Afterwards we must find an API which would allow us to visualize our data across a map.

**Phase 2:** After attaining the necessary developmental environments we will begin implementing our data to be seen across a map along with creating a user interface to interact with the data

**Phase 3:** Both the front and back end development will be done independently from each other, as such once both parts have been completed we move onto joining them together to create a functional final product.

**Phase 4:** After creating a final product we will begin our testing phase. We will create and run testing scripts to check for possible errors. Afterwards we will include any other additional simple improvements that may be requested by the customer or from ourselves to polish off our product

## 7.1 Project organization.

The organization of our team will be split up into 4 separate roles. Though each of the roles will have designated team members, this does not prevent members from temporarily switching roles to assist one another. However, the responsibility for completing each role falls onto its designated member(s).

These roles are:

Front End Developer - Responsible for creating a web application that will allow the user to interact with our gathered data.

Back End Developer - Responsible for creating the database that will implement the Covid Data

Documenter/editor - Responsible for holding important documents such as SDP, SRS, SDD, STP, Covid Data, and any other important documents retaining our project. They will also make necessary edits when needed on existing documents.

Software Tester - Responsible for testing the web app and fixing any incorrect data as well as taking notes for potential quality of life improvements

These roles will work together as a team to complete the project and will produce a final product by the end of phase 4. Both front and back end developers will work independently from one another frome Phase 1-2, and will come back to work together in Phases 3-4. Although all members will work on the maintenance of the final product, the Software Tester will create the testing scripts to find the more general issues while other members search for more unique ones.

The members for each roles are as follows:

| Tom | Front End Developer |
| --- | --- |
| Israel | Front End Developer |
| Duncan | Back End Developer |
| Alex | Back End Developer |
| Safia | Documenter/editor |
| David | Software Tester |

## 7.2 Project Resources

There are three main resources used by this project which include human, data storage, and time. Human resources include our front and back end developers who will create the code for the project, Documenter/editor who will be responsible for all important forms of documentation and data, and Software Tester who will be responsible for maintaining the product. Data storage resources include Google Drive which will hold the majority of our important documentation as well as GitHub which will be where our code is stored and accessed. Time is the final constraint as each member of the team is limited not only from the time provided from the project, but from other classes and responsibilities as well.

We have had no use of public facilities as all aspects of the project were done locally in each individual team member’s home. Team meetings are done virtually due to Covid restrictions. Google Drive and Github were used as our repository for our documentation and database management. This allows easy access anywhere so long as one has access to a computer with a secure internet connection and a Github account.