**COMP-3220 GROUP PROJECT**

**INCEPTION PHASE**

Written By: Team 1

Matthew Muscedere

Hiba Rehman

Karanveer Sidhu

Matt Gagnon

Varun Teja Katakam

Blake Derksen

Tory Provenzano

**Vision and Business Case**

**Vision Case**

1. Purpose:

To create an open data platform for managing and accessing COVID-19 datasets in Canada, supporting public health, research, and development efforts.

2. Scope:

Interactive dashboards to visualize data trends like case numbers, recoveries, and demographics. A data upload system that enables health authorities to upload and update datasets in real time. APIs that allow developers to integrate the data into their apps, offering downloadable datasets in multiple formats.

3. Target Users:

Public health officials to monitor and manage outbreaks. Researchers to access detailed datasets for analysis. Developers to use APIs to build applications. The public to view COVID-19 trends via the dashboard.

4. Key Features:

Real-time visualizations, API access, secure data management, and downloadable datasets.

**Business Case**

1. Problem Statement:

Existing tools for COVID-19 data are limited in scope and lack developer-friendly API access. Public health agencies need a system to manage and distribute data effectively.

2. Value Proposition:

For public health, the system streamlines data collection and improves decision-making. For researchers, it offers easy access to detailed data. For developers, it provides a robust API for creating apps with COVID-19 data. For the public, it offers transparent and up-to-date information on COVID-19.

3. Stakeholders:

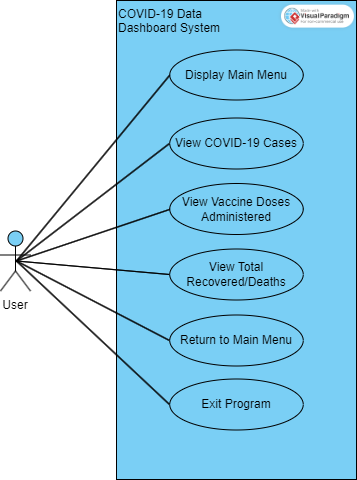
Primary stakeholders include government agencies, researchers, and developers. Secondary stakeholders include the public and non-profits.

4. Business Objectives:

The system improves data access, ensures data accuracy, enhances public transparency, and encourages innovation through open data APIs.

5. Risks:

Data privacy must be ensured by anonymizing and securing sensitive data. The platform must also be scalable to handle growing datasets and user traffic.

**Use-Case Model**

1. **Display Main Menu**

* **Actors:** User
* **Preconditions**: The program is compiled and run/launched by the user.
* **Main Scenario**:
  + The user opens the program.
  + The system initializes and loads the main menu screen.
  + The system displays the main menu options, as followed:
  + "COVID-19 Cases"
  + "Vaccine Doses Administered"
  + "Total Recovered/Total Deaths"
* Each option is a clickable button that leads to a different JFrame when selected.
* **Postconditions**: The main menu is displayed, and the user can select from the options.
* **Alternate Scenarios**:
  + 1a. Loading Delay
    - If the program takes a while to initialize, a loading screen (or icon) is displayed until the main menu is ready.
  + 3a. Failed to Load
    - If the main menu cannot be displayed due to an error, the system displays an error message (with a retry or exit option).

1. **View COVID-19 Cases**

* The user selects the “COVID-19 Cases” button and is given an option to view either “Reported Cases” or “Active Cases”. The system then retrieves the corresponding live dataset from the API and displays it.

1. **View Vaccine Doses Administered**

* The user selects the “Vaccine Doses Administered” button. The system then retrieves the corresponding live dataset from the API and displays it (sorted by dose number or type).

1. **View Total Recovered Cases and Total Deaths**

* The user selects the “Recovered/Deaths” button and is given an option to view either “Total Recovered Cases” or “Total Deaths”. The system then retrieves the corresponding live dataset from the API and displays it.

1. **Return to Main Menu**

* The user selects the “Back” button within the specific frame. The system closes the frame and returns to the main menu.

1. **Exit Program**

* The user selects the “Exit” option, and the system prompts the user with a confirmation dialog box. If the user selects “Yes”, the application is terminated.

**Supplementary Specification**

**Performance**

* **Response Time**: The system should deliver real-time data to users with minimal latency. API responses and dashboard updates must occur within 2-3 seconds under normal traffic conditions.
* **Load Handling**: The platform should support up to 10,000 simultaneous users, including public users, developers accessing the API, and researchers downloading large datasets.
* **Data Refresh Rate**: COVID-19 datasets should be updated at least once per hour to ensure users are accessing the most up-to-date information.

**Scalability**

* The platform must handle increasing amounts of data as COVID-19 cases are reported. This includes a growing dataset for cases, recoveries, and testing rates. It should be designed to accommodate not only current datasets but also potential future expansion.
* The architecture should support horizontal scaling, meaning additional servers or services can be added to handle more users without requiring a full redesign of the system.

**Security**

* **Data Privacy**: All personal information related to COVID-19 cases must be anonymized to comply with Canadian privacy laws (e.g., Personal Information Protection and Electronic Documents Act – PIPEDA).
* **User Authentication**: Public health officials and researchers with upload privileges must have secure login credentials and role-based access control to ensure that only authorized individuals can modify datasets.
* **Encryption**: All data in transit (API calls, data uploads) must be encrypted using SSL/TLS protocols to prevent unauthorized access.

**Usability**

* **User Interface**: The dashboard must be intuitive and accessible to a broad audience, including public health officials, researchers, and the public.
* **Accessibility**: The platform must be compliant with WCAG 2.1 standards to ensure it is accessible to users with disabilities.
* **Error Handling**: In case of failed data uploads or incorrect API usage, the system should provide informative error messages and logs for debugging.

**Availability**

* The system must have a high availability rate, aiming for 99.9% uptime to ensure public health officials and researchers can always access critical data.
* It should have built-in redundancy with backup systems in place to ensure continuous operation even during server failures or maintenance periods.

**Legal and Regulatory Requirements**

* The platform must adhere to Canadian health and privacy laws, ensuring that any data used or shared follows strict guidelines for the protection of sensitive health information.
* Datasets should comply with the Public Health Agency of Canada’s guidelines for COVID-19 data sharing and reporting.

**Maintainability**

* Detailed documentation should be provided for all APIs and key system components to ensure that future developers can easily maintain and enhance the system.

**Localization**

* The platform should support English and French, the two official languages of Canada. All public-facing interfaces, including dashboards, should offer a language toggle option for users.

**Glossary**

**Terminology**

* API: Application Programming Interface
* GUI: Graphical User Interface
* PIPEDA: Personal Information Protection and Electronic Documents Act
* SSL: Secure Socket Layer
* TLS: Transport Layer Security
* WCAG: Web Connect Accessibility Guidelines
* UML: Unified Modelling Language
* Real-Time Data: Information updated instantly for immediate insights.
* COVID-19 Dataset: a collection of information (statistics) related to the pandemic sourced from health organizations.

**Risk List and Management Plan**

**Business Risks:**

* **Stakeholder Misalignment**: Misalignment with expectations of stakeholders (government agencies, public).

**Mitigation**: Regular feedback from stakeholders to ensure alignment.

* **Privacy and Legal Compliance:** Ensuring data privacy and compliance with Canadian privacy laws.

**Mitigation**: Adherence to PIPEDA standards and ensuring data is anonymized.

**Technical Risks:**

* **API Availability/Reliability:** The external API is unable to retrieve real-time data due to some form of downtime.

**Mitigation:** Use cached data or provide the users with a message/notification of the API experiencing downtime.

* **Performance Issues:** The application could have slow loading times or unresponsive frames (especially when retrieving large datasets)

**Mitigation:** Use loading icons to enhance user experience during delays and optimize data retrieval processes.

**Resource Risks:**

* **Development Delays:** Limited time or resources impacting the timely completion of development.

**Mitigation:** Frequent status checks and team adjustments to keep progress on track.

* **Data Quality and Completeness:** Data may be incomplete or inconsistent across sources.

**Mitigation:** Implement data validation checks and provide error handling for incomplete datasets.

**Schedule Risks:**

* **Delays**: Some tasks may depend on other people completing their own tasks before another.

**Mitigation**: Have proper communication between team members, allowing everyone to have the knowledge of if a certain task needs to be completed.

* **Communication:** Poor communication can lead to missing deadlines, redundant work being done, etc.

**Mitigation:** Set a weekly meetup time to where everyone can join, giving us ample time to strategize and work together.

* **Scope Creep:** Having too many features and ideas can increase workload, leading to late submissions.

**Mitigation:** Strictly follow our inception plan to not add too many features, following our elaboration phases to slowly but surely add what we need.

* **Conflicts:** With many people within the team, some people are not available or have time to work on our project, leading to problems with submission and teamwork.

**Mitigation:** Get everyone’s personal schedules to find the best time we can have our weekly meetup, ensure full communication and teamwork amongst each other.

**Prototypes and Proof-of-Concepts**

The prototype of what our application plans to look like is linked below:

[**Link to the Prototype**](https://www.figma.com/design/5TTnX3Cdd9Ur62jh33bOgW/COVID-19-Data-Viewer?node-id=0-1&t=YVrFRsaU0vJxFgPJ-1)

**Iteration Plan**

***Iteration 1: Project Setup & Requirements Gathering***

**1. Project Setup**

* Set up version control on GitHub to organize documentation and collaboration.
* Ensure all team members have access to required resources (dataset links, tools).

**2. Data Exploration**

* Examine the dataset from [COVID-19 GitHub](https://github.com/ishaberry/Covid19Canada) and [Dashboard](https://art-bd.shinyapps.io/covid19canada/) to understand available data types (confirmed cases, hospitalizations, deaths).

**3. Requirements and Feature List Definition**

* Identify key features for the open data management tool (data filters, visualizations).
* Draft a list of system requirements based on available data and envisioned tool functionalities.

**Deliverables:**

* Project vision and scope definition.
* Initial feature list and system requirements.

#### **Iteration 2: Drafting Inception Artifacts & Use Case Diagrams**

**Objective:** Begin drafting the inception artifacts, mapping out project requirements and initial use cases.

**1. Develop Inception Artifacts**

* Draft the nine inception artifacts required for the report, which typically cover areas like vision, scope, stakeholders, risk assessment, and initial requirements.

**2. Use Case Diagram**

* Create initial use case diagrams, identifying actors and high-level use cases (e.g., "View COVID-19 Data," "Filter by Province," "Download Data").
* Outline core interactions for primary functionalities, focusing on user data access and filtering capabilities.

**Deliverables:**

* Drafted inception artifacts.
* Initial use case diagrams with key project interactions.

#### **Iteration 3: Feasibility, Risk Assessment & Initial UML Diagrams**

**Objective:** Assess project feasibility and complete documentation with diagrams and a high-level project plan.

**1. Feasibility & Risk Assessment**

* Evaluate the feasibility of data integration and display features based on available tools and team resources.
* Identify potential project risks (e.g., data updates, handling large datasets) and document mitigation strategies.

**2. UML Diagram Creation**

* Begin a high-level class diagram with core entities (e.g., User, Dataset, Visualization).
* Ensure diagrams reflect the functionality of the open data management tool.

**3. Final Review & Documentation**

* Consolidate inception artifacts, use case and UML diagrams, and other documentation into the final report format.
* Finalize a contribution table showing team member roles and contributions for Phase 1.

**Deliverables:**

* Feasibility and risk assessment document.
* High-level UML diagrams (class and sequence diagrams as applicable).
* Completed inception report (all artifacts and diagrams) in PDF format.

**Phase Plan and Software Development Plan**

**Phase Plan**

Our inception phase, where we brainstorm, setup, gather, and initiate our first tools used, is set to take around 1-2 weeks, ending around October 27th, with polish and submission coming by October 30th.

Our first elaboration phase is due on November 6th, giving us by the time we are done our inception phase one week to complete our first iteration. Most of our effort will be put into making UML diagrams and starting the initial development of our project. We will also have most of the tools (described in Software Development Plan) set up and usable by every member in our group.

Our second elaboration phase is due on November 20th, giving us by the time we are done our first elaboration two weeks to complete it. Our effort will be put into further expanding our project, with 5 or so different use cases implemented, as well as adding/expanding on a visual user interface. We will also likely provide UML diagrams describing all these use cases and have everyone fully setup and implemented in our system of tools.

Our last part of our project is our presentation and report which are due on November 25th and November 27th, respectively. This gives us a week to complete the rest of the project. Our effort will be used making the presentation, which will be used to show off and demonstrate how our project is so far, providing screenshots and explanations to our program. The report will be the culmination of all our work, in a document that explains how we designed, managed, approached, and created the project.

**Software Development Plan**

The applications and tools that we will be using for this project are:

* Team Communication Tools: Microsoft Teams
* Documentation Tool: Microsoft Word
* Prototype Tool: Figma
* Project Management/Tracking: Redmine
* UML Diagram Tool: Visual Paradigm, Lucid Chart
* Code/Document Repository and Version Control: GitHub
* Programming Applications: Visual Studio Code, MyWeb, Overleaf, JFrame
* Presentation Tool: Microsoft PowerPoint

The skills and education required to accomplish this project were provided by the University of Windsor’s Computer Science program. Most of the skills used were from COMP-3220, Object Orientated Software Analysis and Design. This class’s professor is Dr. Ziad Kobti, where he helped answer our questions and lead us in the right direction.

The resources and links to open data case given/used:

* <https://www.cmaj.ca/content/open-access-epidemiological-data-and-interactive-dashboard-monitor-covid-19-outbreak-canada>
* <https://github.com/ishaberry/Covid19Canada>
* <https://art-bd.shinyapps.io/covid19canada/>

**Development Case**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Discipline | Artifact | Inception | Elaboration | | Construction | Transition |
|  | Iteration à | I1 | E1 | E2 | C1 | T1 |
| Business Modeling | Business Case | Must | n/a | n/a | n/a | n/a |
| Requirements | Use-Case Model | Must | n/a | n/a | n/a | n/a |
| Vision Document | Must | n/a | n/a | n/a | n/a |
| Supplementary Specification | Must | n/a | n/a | n/a | n/a |
| Glossary | Must | n/a | n/a | n/a | n/a |
| Risk List | Should | Should | Must | n/a | n/a |
| Management Plan | Must | n/a | n/a | n/a | n/a |
| Iteration Plan | Should | Must | n/a | n/a | n/a |
| Tools List | Must | n/a | n/a | n/a | n/a |
| Design | Design Model | Should | Must | n/a | n/a | n/a |
| Software Architecture Document | Should | Must | n/a | n/a | n/a |
| Prototype/Proof of Concepts | Should | Should | Must | n/a | n/a |
| Implementation | Source Code | Won’t | Won’t | Could | Must | n/a |
| Test | Test Plan | Won’t | Won’t | Won’t | Must | Must |
| Deployment | Executable Software | Won’t | Won’t | Won’t | Could | Must |
| User Manual | Won’t | Won’t | Won’t | Could | Must |
| Project Management | Software Development Plan | Must | n/a | n/a | n/a | n/a |
| Phase Plan | Must | n/a | n/a | n/a | n/a |
| Environment | Development Case | Must | n/a | n/a | n/a | n/a |

Won’t = Artifact will not be worked on in this phase.

Could = Artifact could be worked on in this phase, but not it is necessary.

Should = Artifact should be worked on in this phase, but it is not the priority.

Must = Artifact must be worked on and completed in this phase.

n/a = non-applicable (artifact should have been completed).

**Contribution Table**

|  |  |
| --- | --- |
| Vision and Business Case | Matthew Muscedere |
| Use-Case Model | Hiba Rehman |
| Supplementary Specification | Karanveer Sidhu |
| Glossary | Everyone |
| Risk List and Management Plan | Everyone |
| Prototypes and Proof of Concepts | Matt Gagnon |
| Iteration Plan | Varun Teja Katakam |
| Phase Plan and Software Development Plan | Blake Derksen |
| Development Case | Tory Provenzano |

UML

Class Diagram: <https://lucid.app/lucidchart/03c5b142-7672-4cf5-aa26-cf2c38937654/edit?viewport_loc=-564%2C-1156%2C3103%2C2381%2C0_0&invitationId=inv_f47df850-e6d8-42c5-a901-73e1f36ab4e8>

Sequence Diagram: <https://lucid.app/lucidchart/03c5b142-7672-4cf5-aa26-cf2c38937654/edit?viewport_loc=-1157%2C-566%2C2227%2C1709%2CGATlxTZdFI39&invitationId=inv_f47df850-e6d8-42c5-a901-73e1f36ab4e8>