**Agent Based Covid 19 Vaccine Report Generation**

System Design Document

Table of Contents

**Project Overview2**

Introduction 2

Objectives 2

System Requirements 2

**Design Diagram3**

**Project Design 4**

Access Levels 4

User Interface 4

Login Screen 4

Database Admin Screen 4

Edit User Data Screen 5

Edit Vaccine Data Screen 5

Database Management System 5

Users’ Data Table5

Vaccine Data Table6

Agents 6

**Agent Communication 7**

List of Commands 7

List of Agents 7

Error Codes 8

Command Packet Format 8

**Operation Flow10**

1. **Project Overview**
   1. **Introduction**

COVID-19 vaccines help prevent us from getting infected and protect us from getting severely sick if we do get it. Albertans may wish to prove vaccination status if accessing businesses, other facilities, or travelling. [Alberta vaccine reports](https://www.alberta.ca/covidrecords) will be the only valid Alberta proof of vaccination accepted by operators participating in the [Restrictions Exemption Program](https://www.alberta.ca/covid-19-public-health-actions.aspx#REP) as of November 15. [Valid Alberta vaccine record](https://www.alberta.ca/covid19-vaccine.aspx#records) will show name, birthdate, and vaccination type and date. Our project explains how to use Agent-based model to access and retrieve Covid-19 vaccination reports from the database for people who resides in Alberta.

* 1. **Objectives**

Multi-agent systems technology allows for the development of autonomous software entities (intelligent agents) which are designed to communicate with each other. In our project, we are aiming to have a multi-agent system with the following agents:

**User agent**: Responsible for generating covid report.

**AHS admin agent**: Responsible for find user data/vaccine data and adding vaccine data.

**System database agent**: Responsible for find user data/vaccine data and add/delete/update user/vaccine data.

**Web portal Agent**: Responsible for connecting to other backend agents and get information from them. The above agents will communicate and send messages to each other using Message Protocol. The Gaia Methodology which includes roles, permissions, rights, protocols, and transmission between agents is used for the analysis and design of our application.

The above agents will communicate and send messages to each other using Message Protocol. The Gaia Methodology which includes roles, permissions, rights, protocols, and transmission between agents is used for the analysis and design of our application.

* 1. **System Requirements**

This app is designed using python framework. Our project requires the following:

MySQL(Back-end): is an open-source relational database management system.

Tkinter (front-end): is GUI library to develop User Interface in Python .

PyCharm: PyCharm is a dedicated Python Integrated Development Environment (IDE) providing a wide range of essential tools for Python developers.

SPADE: SPADE software is a middleware for developing and executing multi-agent systems, written in Python.

1. **Design Diagram**

**Database Admin Agent**

**AHS Admin Agent**

**User Agen**t

**Patient/User**

**Permissions/Rights**

**Customer Service Rep- (AHS-811) able to access reports on behalf of patient, send request to database agent for any modifications**

**Representative at the location - AHS or pharmacy -able to enter/ input all info related to patient and vaccination details**

**Full access over the database -Add/Update/Delete patient records and covid reports**

**User-able to enter Patient info –Name, AHNo, Date of Birth on the screen and able to access reports**

1. **Project Design** 
   1. **List of Operation - Access levels**

**Patient/User** – Access and Generate Report (from the front end – screen)

**AHS Admin**- Edit Vaccine and Patient/User Data (from the front end - screen)

**Database Admin** - Update, delete, add patient and covid vaccine records in the database, receives request from User Admin, can view and estimate percentage of the population vaccinated (backend operations)

* 1. **User Interface** 
     1. **Graphical user interface, website

        Description automatically generatedLogin Screen**
     2. **Graphical user interface, website

        Description automatically generatedDatabase Admin Screen**
     3. **Graphical user interface

        Description automatically generatedEdit User Data Screen**
     4. **Graphical user interface

        Description automatically generatedEdit Vaccine Data Screen**
  2. **Database Management**
     1. **Users’ Data Table**

|  |
| --- |
| PatientID–Primary key (Autogenerated) |
| Patient Name |
| AHNo |
| Date of Birth |
| Address |
| Contact |

* + 1. **Vaccine data Table**

|  |
| --- |
| ReportID-Primary key (Autogenerated) |
| Patient Name |
| AHNo |
| Dose 1 Type (Name of Vaccine) |
| Dose 1 Date |
| Dose 1 Address (Pharmacy Location) |
| Dose 2 Type (Name of Vaccine) |
| Dose 2 Date |
| Dose 2 Address (Pharmacy Location) |

* 1. **Agents**

An agent is a software entity that functions continuously and autonomously in a particular environment, often inhabited by other agents and process. In our system there are four agents as follows:

* Web portal Agent
* User Agent
* System Database Agent
* AHS Admin Agent

1. Diagram

   Description automatically generated**Agent Communication** 
   1. **List of Commands**

|  |  |
| --- | --- |
| **Command Name** | **Command ID** |
| UserData | 1 |
| ReportData | 2 |
| UserDataUpdate | 3 |
| UserDataAdd | 4 |
| UserDataRemove | 5 |
| VaccineDataUpdate | 6 |
| VaccineDataAdd | 7 |
| VaccineDataRemove | 8 |
| GenerateReport | 9 |

* 1. **List of Agents**

|  |  |
| --- | --- |
| **Command Name** | **Command ID** |
| Webportal Agent | 1 |
| SystemDatabaseAgent | 2 |
| AHSAdminAgent | 3 |
| UserAgent | 4 |

* 1. **Error Codes**

|  |  |
| --- | --- |
| Success | **0** |
| DataNotFound | 1 |
| DatabaseConnectionFailure | 2 |
| DataUpdateFailed | 3 |
| DataAddFailed | 4 |
| DataDeleteFailed | 5 |

* 1. **Command Packet Format**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sender Agent ID | Receiver Agent ID | Command ID | Error Code | Data |
| 1 Char | 1 Char | 1 Char | 1 Char | X Bytes |

* + 1. **Command UserData**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sender Agent ID | Receiver Agent ID | Command ID | Error Code | Data | Command Type |
| “1” | "2" / "3" | "1" | "0" | ":Name:HCNo" | Request |
| "2" /"3" | "1" | "1" | "0"/"1"/"2" | ":Name:HCNo:DOB:Address:Contact" / "" | Response |

* + 1. **Command ReportData**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sender Agent ID | Receiver Agent ID | Command ID | Error Code | Data | Command Type |
| "1" / "4" | "2" / "3" | "2" | "0" | ":Name:HCNo" | Request |
| "2" /"3" | "1" / "4" | "2" | "0"/"1"/"2" | ":Name:HCNo:Dose1Type:Dose1Date:Dose1Address:Dose2Type:Dose2Date:Dose2Address" / "" | Response |

* + 1. **Command UserDataUpdate**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sender Agent ID | Receiver Agent ID | Command ID | Error Code | Data | Command Type |
| "1" | "2" | "3" | "0" | ":Name:HCNo:DOB:Address:Contact" | Request |
| "2" | "1" | "3" | "0"/"1"/"2"/”3” | "" | Response |

* + 1. **Command UserDataAdd**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sender Agent ID | Receiver Agent ID | Command ID | Error Code | Data | Command Type |
| "1" | "2" | "4" | "0" | ":Name:HCNo:DOB:Address:Contact" | Request |
| "2" | "1" | "4" | "0"/"2"/”4” | "" | Response |

* + 1. **Command UserDataDelete**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sender Agent ID | Receiver Agent ID | Command ID | Error Code | Data | Command Type |
| "1" | "2" | "5" | "0" | ":Name:HCNo" | Request |
| "2" | "1" | "5" | "0"/"1"/"2"/”5” | "" | Response |

* + 1. **Command VaccineDataUpdate**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sender Agent ID | Receiver Agent ID | Command ID | Error Code | Data | Command Type |
| "1" | "2" | "6" | "0" | ":Name:HCNo:Dose1Type:Dose1Date:Dose1Address:Dose2Type:Dose2Date:Dose2Address" | Request |
| "2" | "1" | "6" | "0"/"1"/"2"/”3” | "" | Response |

* + 1. **Command VaccineDataAdd**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sender Agent ID | Receiver Agent ID | Command ID | Error Code | Data | Command Type |
| "1" | "2" /"3" | "7" | "0" | ":Name:HCNo:Dose1Type:Dose1Date:Dose1Address:Dose2Type:Dose2Date:Dose2Address" | Request |
| "2" /"3" | "1" | "7" | "0"/"1"/"2"/”4” | "" | Response |

* + 1. **Command VaccineDataDelete**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sender Agent ID | Receiver Agent ID | Command ID | Error Code | Data | Command Type |
| "1" | "2" | "8" | "0" | ":Name:HCNo" | Request |
| "2" | "1" | "8" | "0"/"1"/"2"/”5” | "" | Response |

* + 1. **Command GenerateReport**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sender Agent ID | Receiver Agent ID | Command ID | Error Code | Data | Command Type |
| "1" | "4" | "9" | "0" | ":Name:HCNo" | Request |
| "4" | "1" | "9" | "0"/"1"/"2" | Hex Bytes of Generated PDF | Response |

1. **Operation Flow**

* Patient Enters Patient details on the form/screen (Name, AHNo, DateofBirth)
* The information is transferred from the sender to the receiver agent through XMPP based messaging protocol (Spade framework)
* This request is sent to the database and the search query is executed against the database (MySQL workbench).
* The results are fetched from the database and the response is received back through the agents
* The final output is displayed on the screen/form