Project Name:

Online Social Insurance Number (SIN) generation system.

Requirements:

There are 5 actors - agents that are required in the project as given below:

- User Agent
- Verification Agent
- Authorization Agent
- Database Agent
- SIN Generator Agent

Scope:

The scope of this project is to create a web application for immigrants and citizens to generate their Social Insurance Number (SIN) online instead of physically going to Service Canada. Any person with valid documents can do the SIN generation process online within Canada.

Functionalities of individual agents and workflow:

- Web Portal to raise an application for SIN generation.
- A user who can login to the SIN generator portal - web portal using Email id and Date of Birth and password.
- Once the user enters the required information on the screen, this request being transferred to the backend to create a JWT (Json Web Token) token using authorization agent
- After authorization, it lands on another web page to submit the SIN application along with necessary personal information I.e., passport number, study permit number, and study permit expiry date.
- Using the above provided personal information, SIN will be generated with the help of SIN agent.
- The generated SIN along with the passport number and an expiry date will be saved in the database agent.
- After successful insert/update, SIN pdf is displayed on the screen and the user can download the pdf from the portal.

What We Will Be Using:

- Python 3.7
- Tkinter (For GUI)
- Spade (Multi Agent System Framework)
- MongoDB (Database Management)
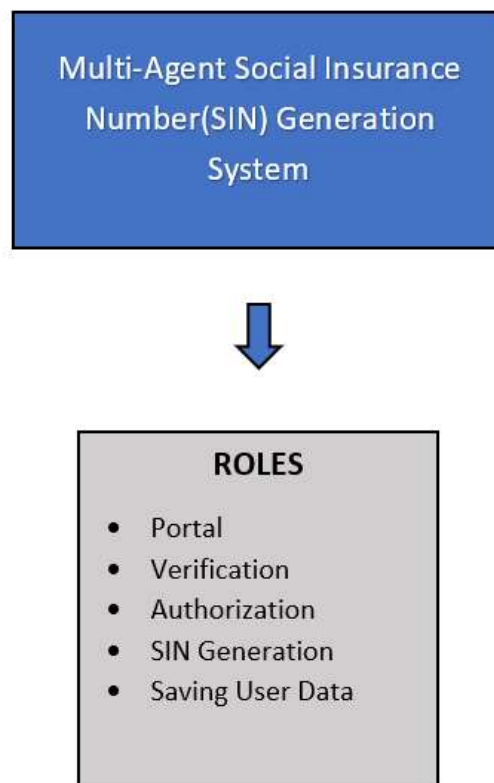- fpdf (PDF Report Generation)
- Pillow (Documents Operation)

Agents and their responsibilities:

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:
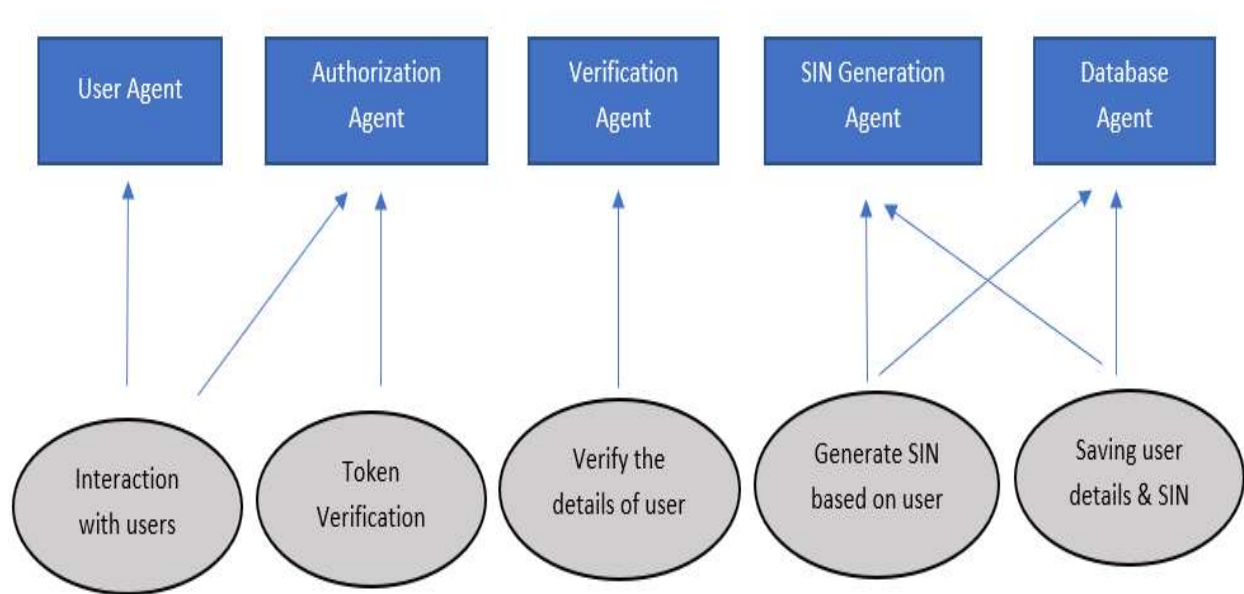
- User Agent – Responsible for taking input from the user such given name, last name, email address, Passport number, permit number, permit expiry date, DOB, etc.
- Verification Agent – Responsible for verifying the details of the user like passport number, permit expiry date, permit number, etc.
- Authorization Agent – Responsible for generation of a JWT (JSON Web Token) which will be used for authentication during login or signup of the user.
- Database Agent – Responsible for handling database related CRUD (Add/Update/Delete) operations.
- SIN Generator Agent – Responsible for generating Social Service Number.

Agent System Architecture:

Roles Model:

Multi-Agent Social Insurance Number(SIN) Generation System

ROLES

- Portal
- Verification
- Authorization
- SIN Generation
- Saving User Data

Agents Model:

User Agent	Authorization Agent	Verification Agent	SIN Generation Agent	Database Agent

Interaction with users

Token Verification

Verify the details of user

Generate SIN based on user

Saving user details & SIN

Role Schemas:

Role Schemas	Database
Description	To store a user token and SIN number based on the user's email id, passport, study permit. Number/expiry date, DOB, and password
Protocols and Activities	UserDataAdd, UserDataUpdate, UserDataRemove, SendUserData, SendResponse
Permissions	Read: UserDatabase : user information from web portal Write: AuthDatabase: storing Json web token SINDatase : storing generated SIN number
Responsibilities	Liveness: SHOW_USER_DATA = Geuserinfo.sendUserData ADD_USER = getUserinfo.AdduserData.sendresponse ADD_SIN = get.SININfo.AddSINnumber.sendresponse Safety: Successful connection with web portal

Role Schemas	Portal
Description	Takes input from the user's information to create the SIN number and generates and displays SIN number on the screen
Protocols and Activities	GetUserInfo, RequestSIN, GenerateSIN, DIsplaySIN
Permissions	Read: Information from the user Write: Display data on the portal

Responsibilities	Liveness : LOGIN = GetloginCredentials.verifylogincredentilas DISPLAY_SIN_NUMBER (showSinNumber) = getSininfo.RequestSinNumber Safety : successful connection with database

Role Schemas	Authorization
Description	To generate a user token based on user input and to check if the token generated is valid or not
Protocols and Activities	GenerateToken, CreateUser, ValidateUser, AuthorizeUser
Permissions	Write: AuthDatabase: Storing token in the database Read: Information from the user
Responsibilities	Liveness property – TokenService = UserData.GenerateToken SignUpService = AddUser.CreateUser LoginService = ValidateUser.Authorize Safety: Successful and secure connection with database agent and web portal.

Role Schemas	SIN Generator
Description	To receive input from user agent and based on data like Passport number, permit expiry date, etc. generate a SIN number for the applicant.
Protocols and Activities	GenerateSIN, AddSINData, UpdateSINData, SendResponse, DisplaySIN
Permissions	Write: SINDatabase: storing SIN generated in the database
Responsibilities	Liveness property – SINGenerationService = CreateSIN.AddData

	DisplayService = UserInfo.DisplaySIN Safety: Successful and secure connection with database agent and web portal.

| Role Schemas	Verification
Description	Based on the input received from the user agent the details entered by the user will be verified. The details will also be verified with the database agent if a SIN already exists for the applicant or not.
Protocols and Activities	VerifyData, ReadUserData, ValidateSINExists, SendResponse
Permissions	Write: UserDatabase: Storing user data in the database Read: Information from the user and UserDatabase
Responsibilities	Liveness property – VerificationService = UserData.VerifyDetails CheckSINService = GetUserInfo.ValidateSINExists Safety: Successful and secure connection with database agent.

Interaction Model:

| Protocol	User	Verification	Authorization	Database	SIN Generator
Purpose/Parameters	Collect user data and send it for verification	Verify the user details and store them in database	Generate the JWT token for authorization	Handle database related CRUD operations	generate SIN based on the user information
Initiator(s)	portal	Backend Server	Backend Server	Backend server	Backend Server
Receiver(s)	Verification Agent	Database Agent	Database Agent	SIN Generator Agent	Portal
Processing	User data transfer to verification agent	Verify and approve the user details	Generation of the JWT token	Add/Update/Delete operations	Generation of SIN

Services Model:

Service	Inputs	Outputs	Preconditions	Postconditions
Interaction with User	Email, Date of Birth, and password.	Web form page for SIN application.	Web portal instance is created and associated with the user.	Request authorization agent to create token.
Token Verification	Email, Date of Birth, and password.	JWT (Json Web Token) token	Sends portal inputs to authorization agent	User Enters personal information in form
Verify Details of User	User inputs Name, Passport Number and Permit information	Success message shown upon valid field entry.	Web form instance created asks for personal details.	The field data is sent into the database agent
Generate SIN based on user	Create SIN based on input received from the user agent	Generate 9-digit SIN Number	Database agent verifies entries to be unique and valid.	SIN displayed on screen.
Saving User Details and SIN	Request from Verification Agent to update database.	Sends success/failure message for operation types Add/Remove/Update.	Web form instance created asks for personal details.	Sends request to SIN agent for generating a 9-digit number

Acquaintances Model:
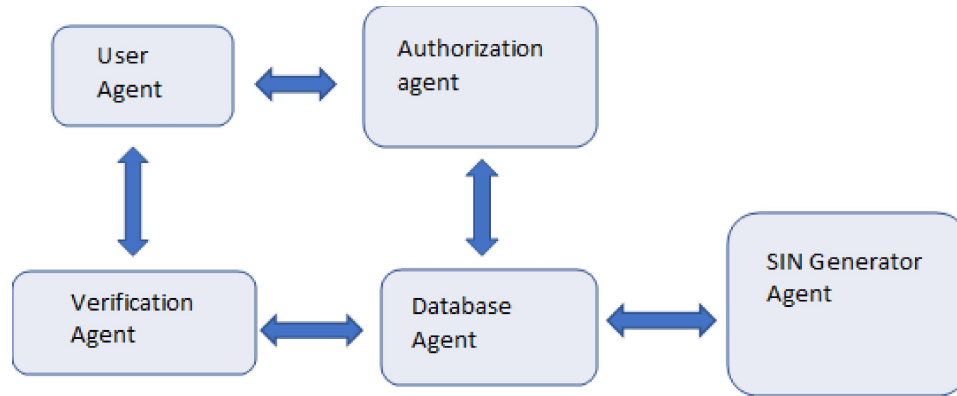
Part 2: Detailed Development Document

1. Message sequence chart (i.e. interactions and protocols between agents):
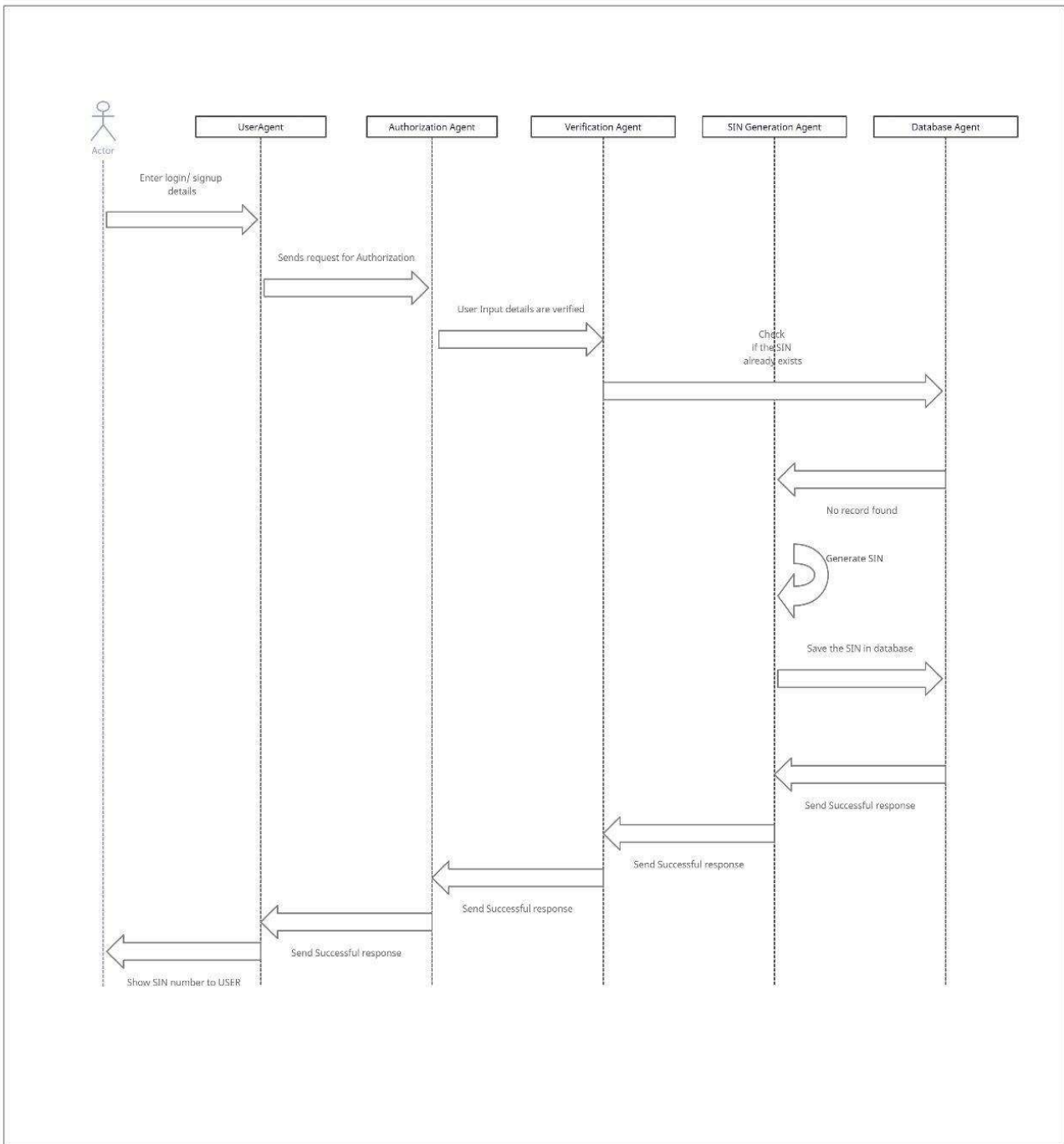
2. Use Case Definitions (for all agents):

- USER AGENT

Brief Description	Handles the interface and GUI of the overall system
Precondition	GUI has been loaded
Postcondition	Results are showing
Process Steps:	
1	Sends user input to the Authorization Agent
2	Receives SIN Number from the SIN AGENT and sends it to GUI

- AUTHORIZATION AGENT

Brief Description	Handles the authorization of the application, creating JSON WEB Token
Precondition	Users have input required details for creating JSON web token
Postcondition	JSON web token (JWT) is created and sent to the database agent
Process Steps:	
1	generates the JSON web token based on the users' details
2	Sends it to the database agent

- VERIFICATION AGENT

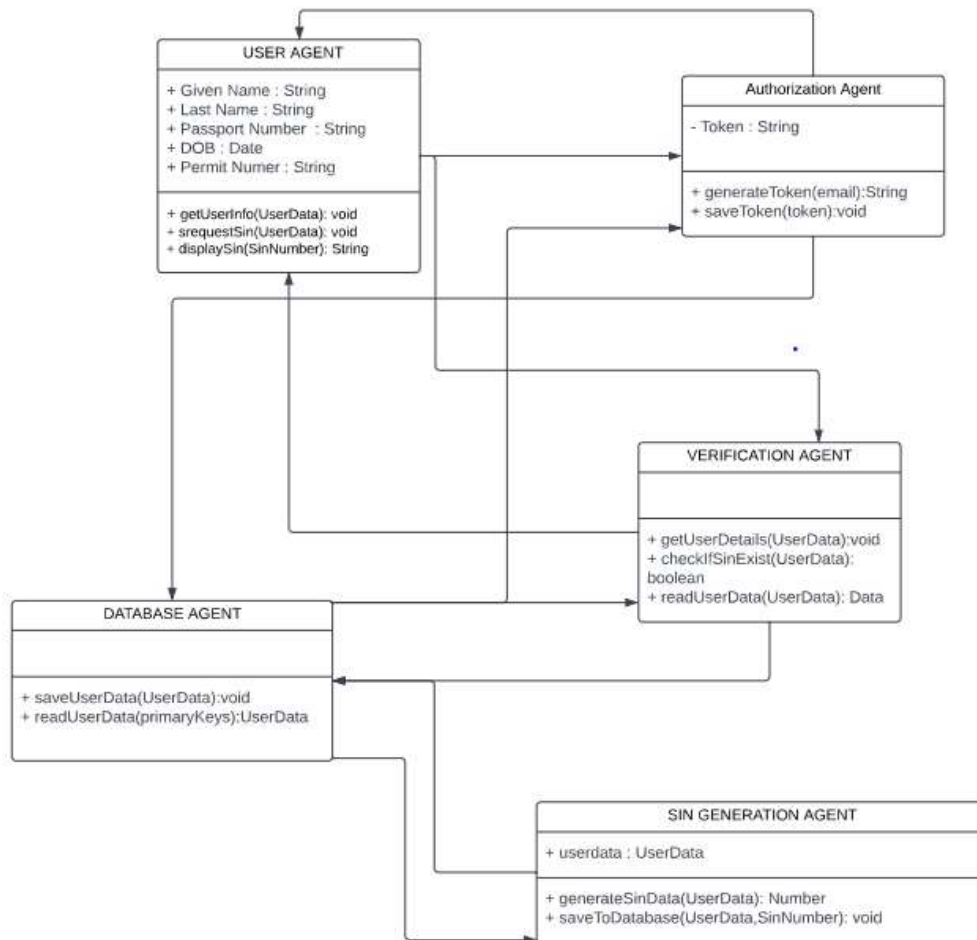| Brief Description	Handles the users' data verification that include passport number, study permit expiry date, permit number, etc.
Precondition	Users have correctly input the required details
Postcondition	Verified the user input details and checks whether SIN number exists or not
Process Steps:	
1	Query the database and check on the users' social insurance number
2	Receives the response from the database agent, if the SIN number already exists or not

- SIN GENERATION AGENT

Brief Description	Responsible for generating SOCIAL INSURANCE NUMBER
Precondition	Database agent verify entries to be unique and valid
Postcondition	Successfully generates the social insurance number
Process Steps:	
1	Generates the unique number (SIN)
2	Sends to database agent for saving it into the database

- DATABASE AGENT

Brief Description	Handles the database based CRUD (create, add, delete) operations
Precondition	Web form instance created and asked for personal details
Postcondition	Sends a request to SIN generation agent for generating 9-digit number
Process Steps:	
1	Get the data from the verification agent and saves it in the MongoDB
2	Reads the data to check whether users' data exist or not and sends ack to the verification agent
3	Saves the SIN number to the database

Class Diagram:

USER AGENT

+ Given Name : String
+ Last Name : String
+ Passport Number : String
+ DOB : Date
+ Permit Numer : String

+ getUserInfo(UserData): void
+ srequestSin(UserData): void
+ displaySin(SinNumber): String

Authorization Agent

- Token : String

+ generateToken(email):String
+ saveToken(token):void

VERIFICATION AGENT

+ getUserDetails(UserData):void
+ checkIfSinExist(UserData):
boolean
+ readUserData(UserData): Data

DATABASE AGENT

+ saveUserData(UserData):void
+ readUserData(primaryKeys):UserData

SIN GENERATION AGENT

+ userdata : UserData

+ generateSinData(UserData): Number
+ saveToDatabase(UserData,SinNumber): void

References:

https://github.com/vyasdeep668/Agent-Based-Covid19-Database-Management-System