



CLiGO SYSTEM DESIGN

Output Design:

This system is an SMS based system, so the output is going to be an SMS text message which would be sent to the subscribers of the system. Since it is a text message, there is not form of formatting needed but the only precaution to be taken here is for each text not to be more than 160 characters. With this, any mobile phone can be used to subscribe and receive text messages. An example would be “Take foods that give folic acid such as spinach, and orange juice, to help protect you and your child from anemia.”

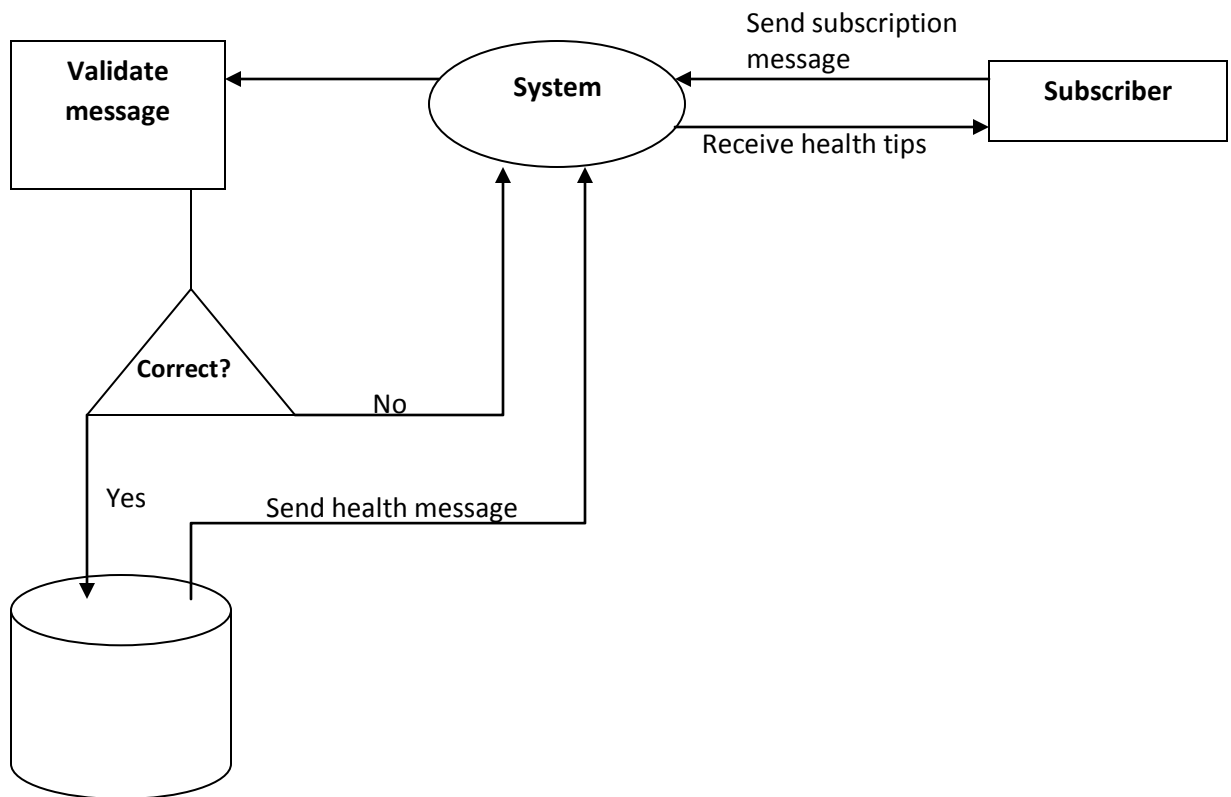
Input Design:

For the administrator side, to be able to feed the required data into the system, a web interface would be developed using python with django. This interface would have fields for the message, message type and the specific period of pregnancy the message is meant for. Another web interface would be created to be able to add hospitals into the system, each of which would be assigned a unique code. The messages would be of three types; general messages, dietary messages and precautionary messages.

The user of the system has to subscribe to be able to receive messages. To subscribe, the user has to type her firstname, number of weeks pregnant and the hospital’s unique code, to a short code. The hospital’s unique code would be found on the posters, customized for each hospital. A user subscription would take a form as in the following example; “Akos, 6, 3456”

For a user to unsubscribe to the system, she has to send the special word “out” to the shortcode, where her information would be removed from the recipient’s database.

Dataflow Diagram:



Functional Models

The functional models describe the business processes and the interaction of the system with its environments. The models identified in this system are subscription and message entry.

USE CASE NAME: Subscription		ID: 1	IMPORTANCE: High
PRIMARY ACTOR: End user		USECASETYPE: Detail	
STAKEHOLDERS End users want to subscribe to receive health tips.			
BRIEF DESCRIPTION: This use case describes how a end user subscribes and is registered into the system			
TRIGGER: when the user visits the hospital and is informed of her pregnancy by a medical doctor			
TYPE: External			

Table 1. Use case description for end user subscription

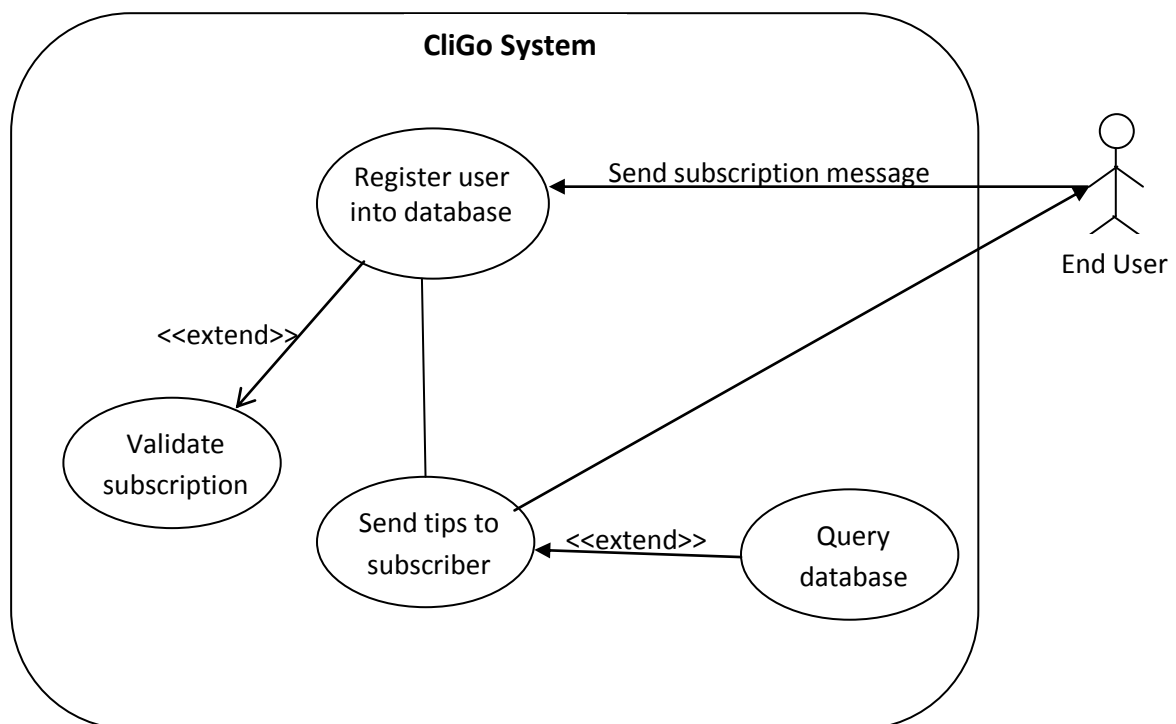


Figure 1. Use case diagram for end user subscription

USE CASE NAME: Message entry	ID: 2	IMPORTANCE: High
PRIMARY ACTOR: Administrator	USECASETYPE: Detail	
STAKEHOLDERS Administrator wants to update the messages in the database		
BRIEF DESCRIPTION: This use case describes how a end user subscribes and is registered into the system		
TRIGGER: when administrator comes across new information during research and wants to update the database.		
TYPE: Internal		

Table 2. Use case description for message entry

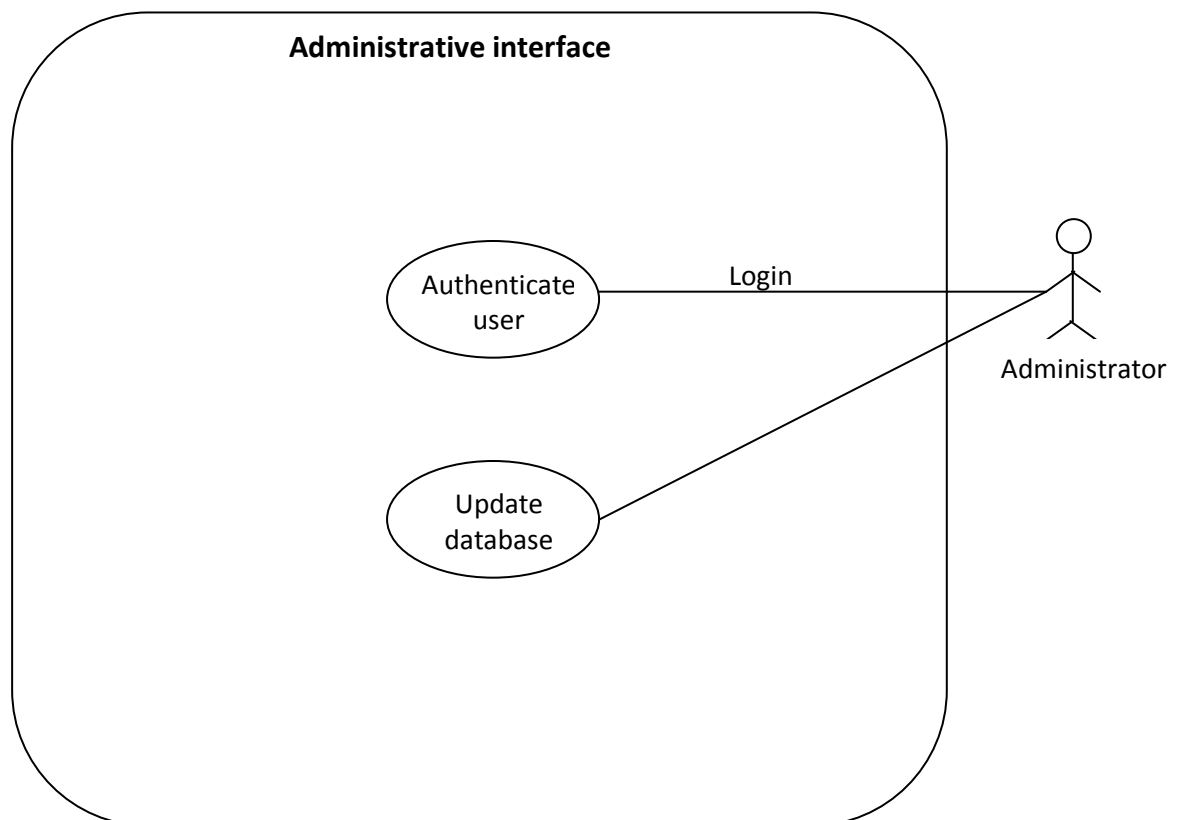


Figure 2. Use case diagram for message entry

Database design:

The database engine used is sqlite3 since it is quite easy to use with django. It would be used to hold information on all users of the system and the health tips to be sent. The database is made up of the following tables:

- Hospital
- Subscriber
- Messages
- Sent_messages

Hospital

Name	String
CenterCode	numeric
Contact number	String
Location	string

Subscriber

Number_of_weeks	Numeric
Name	String
Telephone_number	Numeric
CenterCode	Numeric

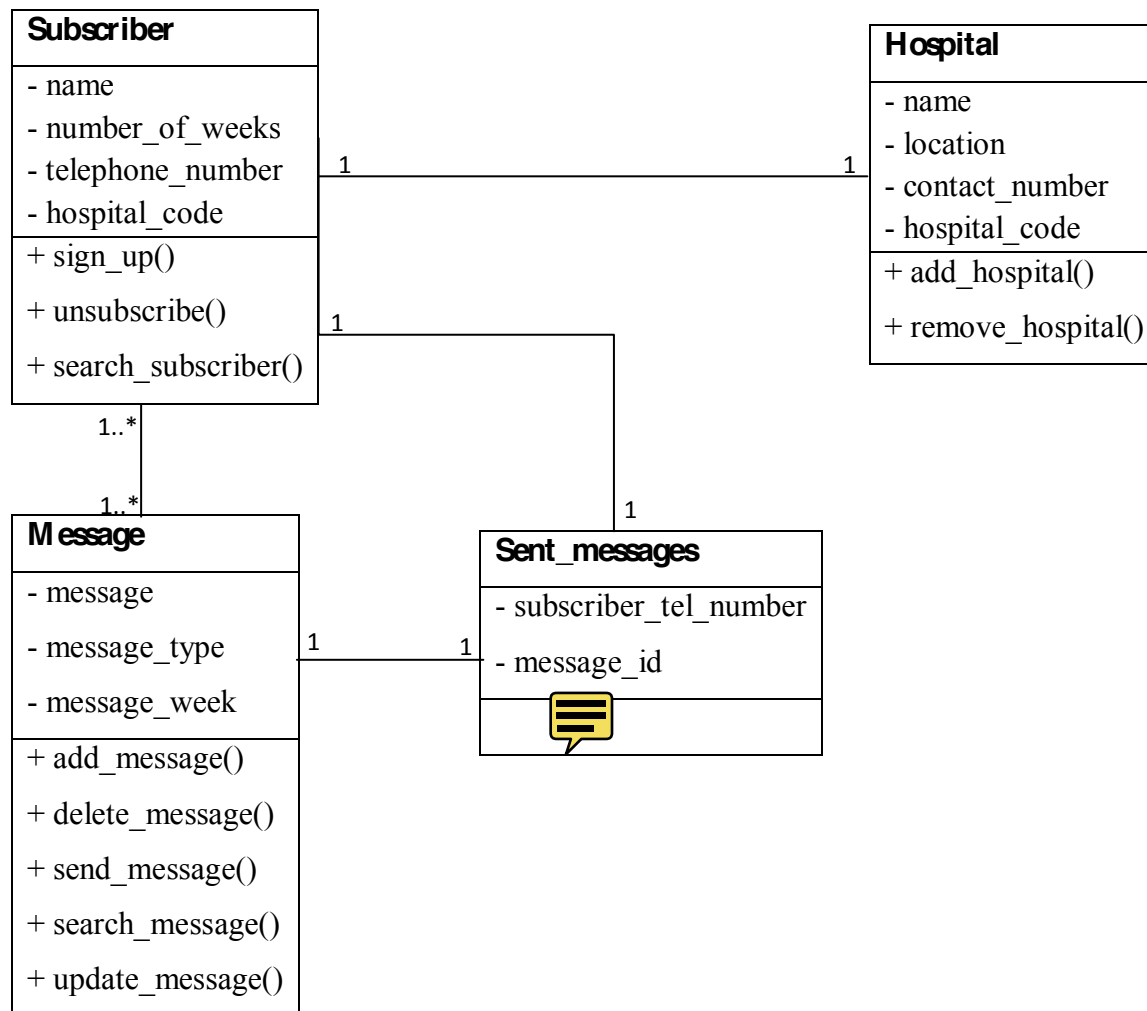
Messages

message	String
Message_type	String
Message_week	numeric


Sent_message

Subscriber_tel_number	Numeric
Message_ID	Numeric

Class diagrams:



Data Validation and verification:

Data captured from the subscriber is supposed to be of the correct format before being stored into the system. To do this, we employed data validation and verification methods, to validate the authenticity of the subscription message. If a subscriber sends an empty text message, or a message with an incorrect format, a response message is sent back advising her to correct the mis On the front end of the system, where the messages are entered into the system, a method to check the length of the input message would be implemented. This is to make sure that the system does not take in any message more than 160 characters.

Security design:

Django already provides us with the administrator user interface, where a user can be created and assigned administrative roles or other user-specific roles. The main mode of security is the use of passwords. Since it is a web based SMS system, any privileged user would need a password with specifically defined privileges to login to the system.