Progetto Maturità Bot Telegram

Covid-19

Link al bot: <https://telegram.me/corona_crawler19_bot>

Link al repository github: <https://github.com/SteCodeLevi/botcovid19>

**Scopo del progetto**

La nostra idea iniziale era quella di creare un bot Telegram che permettesse agli utenti di poter ottenere attraverso dei semplici comandi informazioni aggiornate sui dati dei contagi e notizie odierne per continente e a livello regionale per l’Italia.

**Cosa è un bot?**

Il termine bot è l’abbreviazione di robot e si riferisce a un account aggiuntivo, una sorta di utente non umano appunto, che è capace di **compiere azioni in modo automatico**. Telegram una delle più note applicazioni di messaggistica permette di crearli e di programmarli perchè svolgano specifiche funzioni

**Creazione del bot su Telegram cose necessarie**



1. Inizialmente abbiamo scritto a [**Botfather**](https://telegram.me/BotFather)**,**il bot di Telegram dedicato proprio alla creazione dei nuovi bot. Nella chat abbiamo utilizzato il comando **/newbot** per cominciarne la creazione. Botfather chiede il **nome** da assegnare al bot, poi lo **username**.Dopo questi passaggi abbiamo ricevuto un token. Ogni bot di Telegram è dotato di un **token**, ovvero di una sequenza composta da numeri e da lettere che identifica in maniera univoca il bot e che è utilizzata dalle Web-app e dalle API per richiamare, appunto, quello specifico bot. Questo token permette di mettere in funzione il bot sia, successivamente, di controllarlo.
2. Successivamente ciò di cui avevamo bisogno era uno spazio web dedicato che ci permettesse attraverso il linguaggio PHP da noi scelto di gestire i comandi del bot e integrarli alle interrogazioni attraverso le API di Telegram. Il servizio di hosting che abbiamo utilizzato ci è stato fornito da <https://www.netsons.com/> che mette a disposizione un web server, una base di dati un servizio ftp per il trasferimento dei file ed altri servizi che però non abbiamo utilizzato massivamente per questo progetto. Da notare che per funzionare il servizio di hosting deve utilizzare il protocollo HTTPS per poter comunicare con Telegram altrimenti non funziona.
3. **Tutte le richieste formulate al bot**seguono questo schema: https://api.telegram.org/botTOKEN/METHOD\_NAME, dove al posto della parola Token abbiamo inserito il token fornitoci da BotFather.

A questo punto ci sono due metodi per ottenere i messaggi che gli utenti inviano al bot :

* Tramite la tecnica di long polling
* Tramite la tecnica webhook

La tecnica long polling consiste nell’interrogare periodicamente i server di Telegram per determinare se ci sono messaggi da processare.

Tramite la tecnica webhook invece riceveremo automaticamente su una “callback” i nuovi messaggi da processare direttamente dai server di Telegram. Attraverso il long polling è il nostro codice che periodicamente deve interrogare i server di Telegram per ottenere nuove informazioni, avviene invece il contrario tramite l’uso di un webhook quindi per semplicità abbiamo deciso di utilizzare questo metodo.

Tutto quello che abbiamo fatto è stato chiamare il metodo setWebHook della Bot API tramite il seguente URL:

https://api.telegram.org/bot{my\_bot\_token}/setWebhook?url={url\_to\_send\_updates\_to}

per testare che il bot funzionasse e tutta la procedura iniziale fosse corretta abbiamo scritto questo semplice codice PHP che permette al bot di rispondere a chi gli scrive ciao con Ciao e il suo nome utente



In seguito a ciò abbiamo iniziato a progettare tutta la parte officiale che ci serviva per il proggetto.

Ciò che dovevamo realizzare per recuperare i dati sui contagi e le notizie aggiornate è ciò che viene detto in gergo informatico il **web scraping** ovvero l’estrazionedi dati da un sito web in modo automatico attraverso degli opportuni script realizzati in PHP e il supporto della libreria [**PHP Simple HTML DOM Parser**](https://simplehtmldom.sourceforge.io/)che permette di effettuare il DOM parsing cioè estrarre i principali elementi di una pagina web che ci servivano.

Inanzitutto abbiamo deciso da quali pagine web avremmo estrapolato i dati sui contagi e le notizie e questi sono stati i siti scelti perchè li reputavamo i più affidabili.

Dati contagi mondiali

<https://www.worldometers.info/coronavirus/>

Dati contagi Italia per regioni

<https://github.com/pcm-dpc/COVID-19/blob/master/dati-regioni/dpc-covid19-ita-regioni-latest.csv>

Notizie per continente

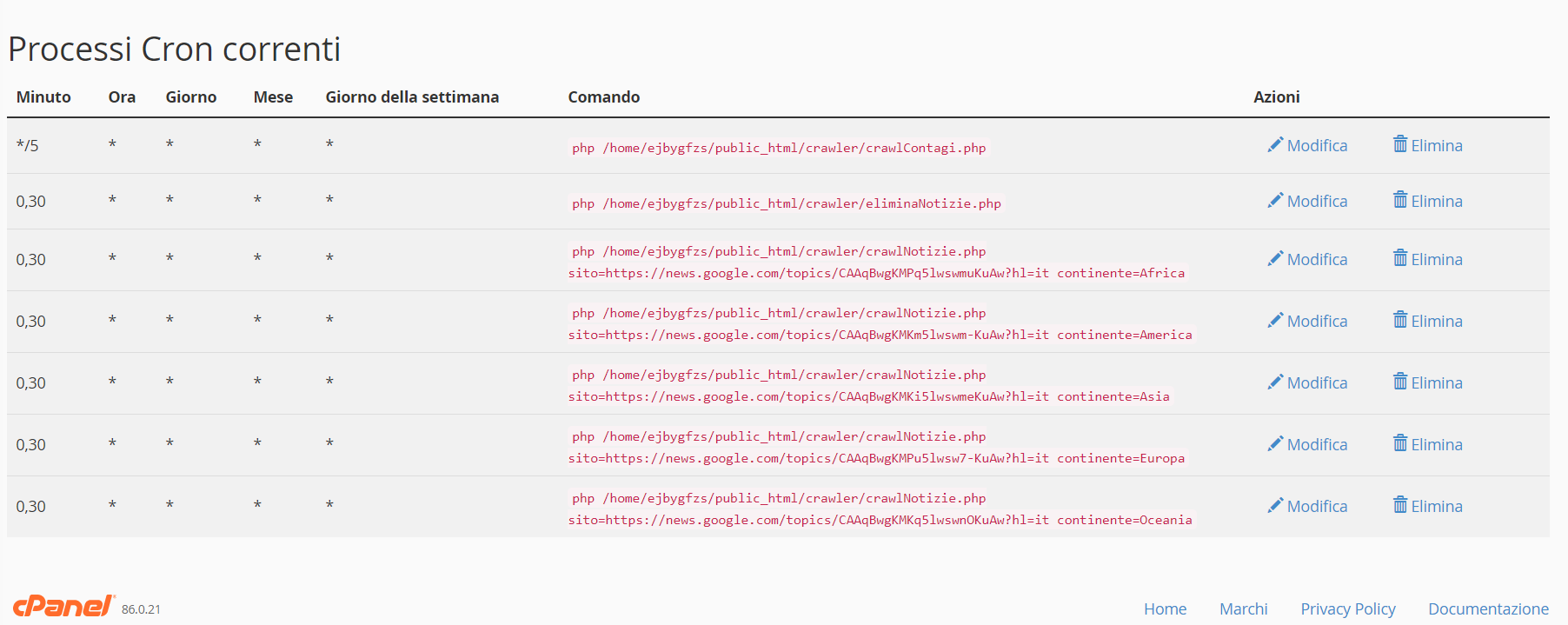
<https://news.google.com/topics/CAAqIggKIhxDQkFTRHdvSkwyMHZNREZqY0hsNUVnSnBkQ2dBUAE/sections/CAQqEAgAKgcICjD2uZcLMJbirgMwwrvMBg?hl=it&gl=IT&ceid=IT%3Ait>

**Come funziona il nostro crawler web per il recupero dei dati**

Il nostro crawler web è composto principalmente da 2 script php che si occupano di recuperare i dati e sono crawler.php che al suo interno contiene due funzioni statiche che permettono di ottenere i dati sui contagi per l’italia e a livello mondiale mentre invece crawlNotizie.php recupera le notizie per continente mediante un link passato in get e il continente associato

Vedi file <https://github.com/SteCodeLevi/botcovid19/tree/master/crawler>

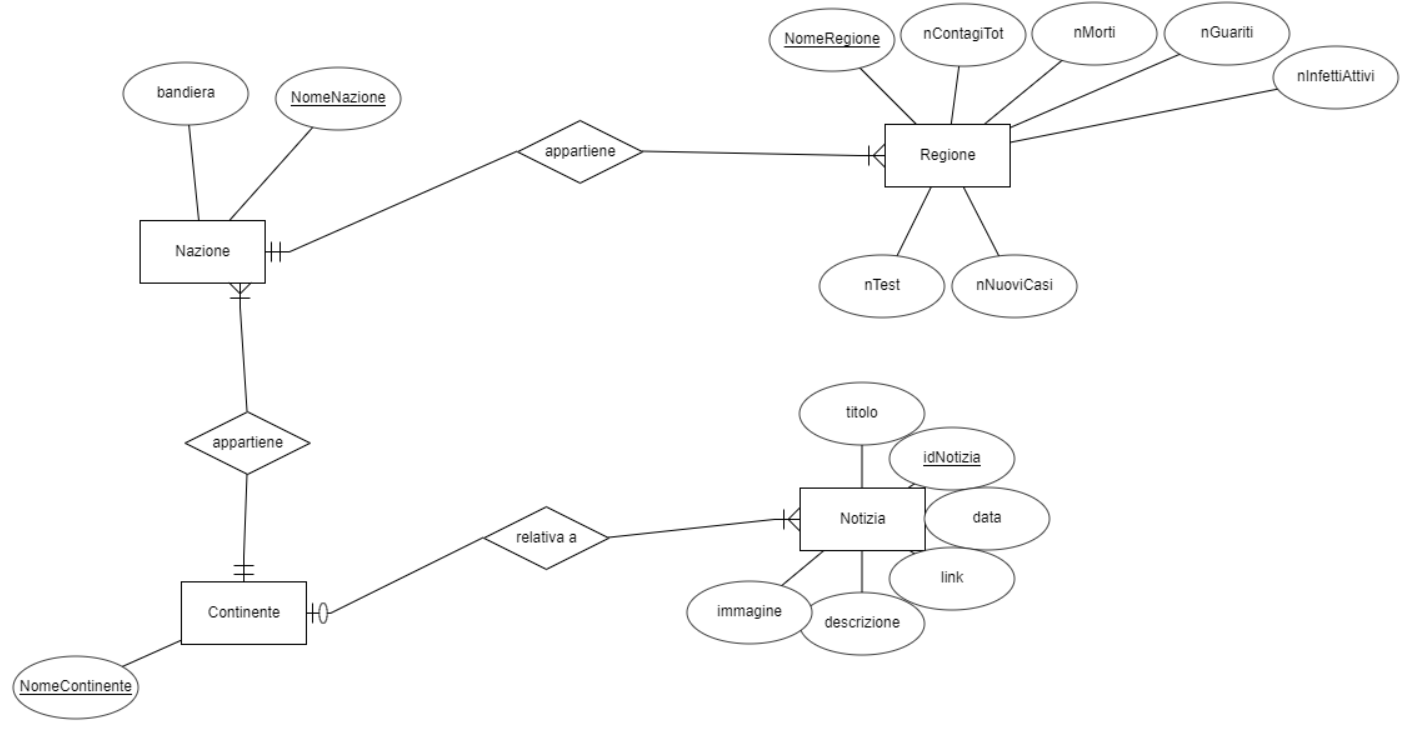
Tutti questi script per il recupero dei dati sono gestiti mediante dei processi di Cron che in modo automatico vengono eseguiti e richiamano gli script circa ogni mezz’ora per tenere aggiornati i vari dati e notizie



Come si può vedere dalla foto le notizie vengono aggiornate ogni 30 minuti e eliminate dal database mentre i dati sui contagi vengono semplicemente sovrascritti ogni 5 minuti perchè cambiano più rapidamente, inoltre si può notare come scritto prima che lo script crawlNotizie.php viene eseguito passando in get il nome del sito da cui estrapolare i dati e il nome del corrispettivo continente

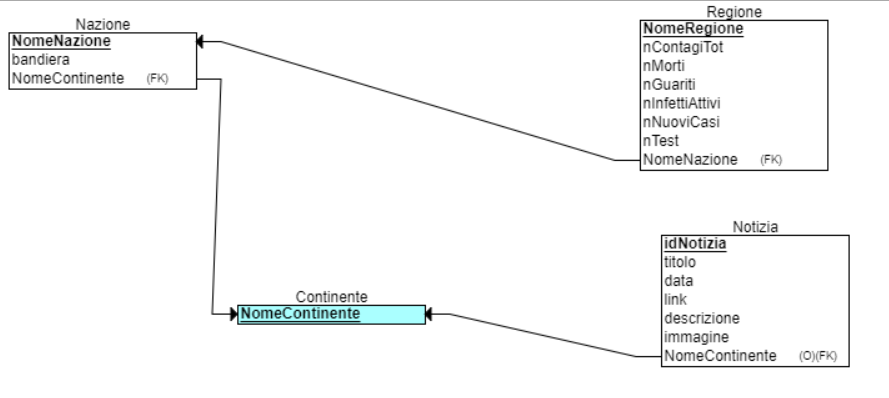
**Dove vengono salvati i dati?**

Abbiamo deciso per comodità di salvare tutti i dati sul database messoci a disposizione dal sito di hosting, così il nostro bot Telegram interroga il database e reperisce i dati molto facilmente.



Lo schema ER qui sopra rappresenta la nostra base di dati nella quale sono presenti come entità Continente, Nazione, Regione e Notizia. I dati su contagi sono contenuti tutti nell’entità Regione che nel caso dei contagi per Nazioni prende il nome stesso della Nazione esempio: Francia-Nazione Francia-Regione mentre solamente per l’Italia possiede le regioni con i dati diversi por ognuna di esse.

Mentre per le Notizie le abbiamo rilegate ad un sigolo continente così che possiamo suddividerle



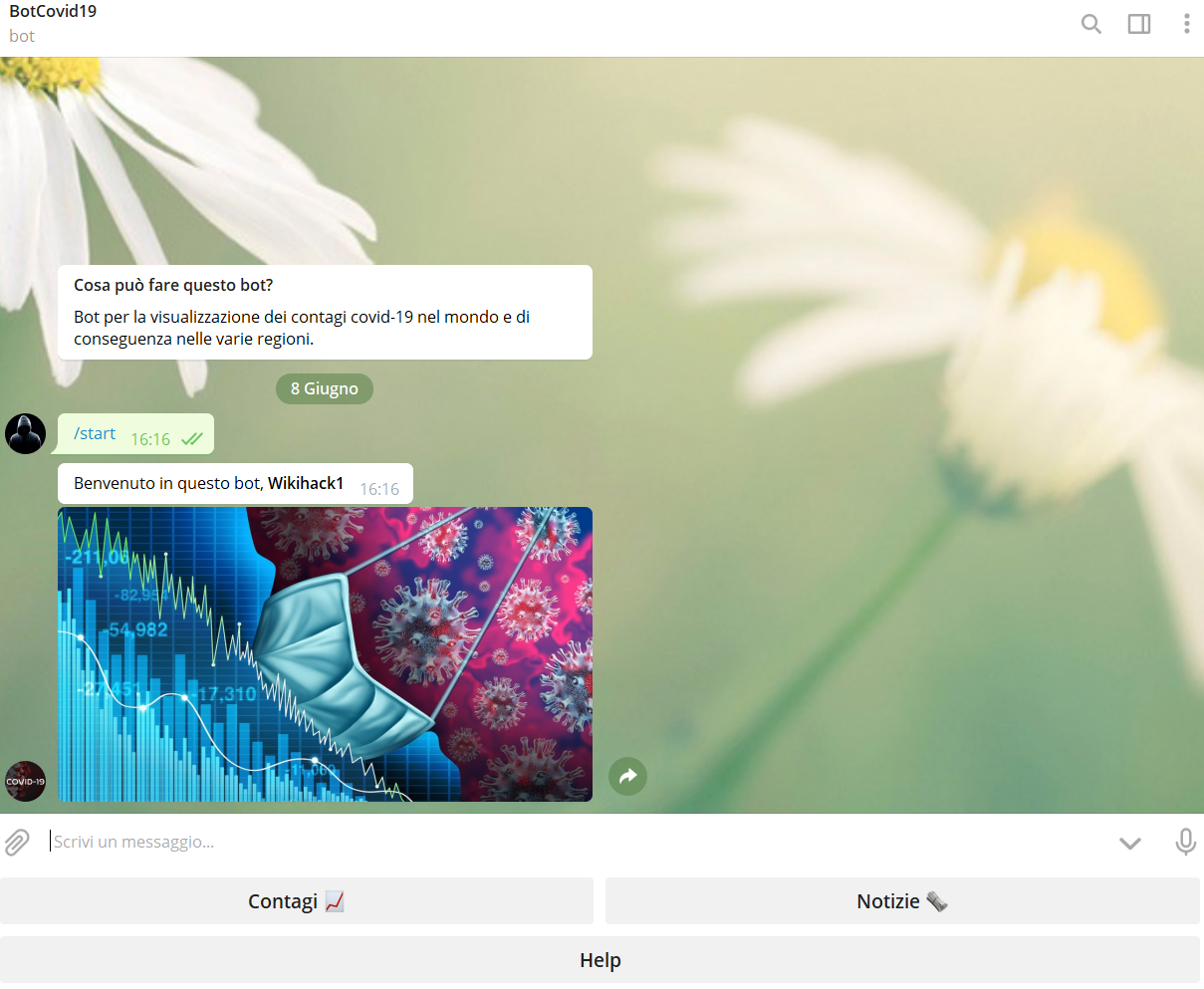
Per vedere le istruzioni sql per la creazione del database vedi <https://github.com/SteCodeLevi/botcovid19/blob/master/documentazione/botCovid19.sql>

Ovviamente nella parte di codifica del crawler php abbiamo implementato delle classi DAO(Data Access Object) permettono le operazioni di CRUD e DO(Data Object) che codificano una tupla delle tabelle come un oggetto e ne rendono più facile la gestione

Vedi datamodel <https://github.com/SteCodeLevi/botcovid19/tree/master/Datamodel>

In questo modo riusciamo ad interagire con il nostro database ad oggetti e utilizziamo una connessione di tipo PDO implementata nella classe database.php .

Una volta Terminata tutta questa parte di codifica per il recupero dei diversi dati e salvataggio su database ci siamo infine dedicati alla preparazione del mockup del bot su Telegram che dispone principalmente di due funzioni mostra contagi e mostra notizie raggiungibile attraverso dei pulsanti.



La parte di codice php che gestisce tutto il bot è contenuta nel seguente file index.php

<https://github.com/SteCodeLevi/botcovid19/blob/master/botcovid19Telegram/index.php>

al suo interno abbiamo creato delle funzioni che permettono l’invio dei contagi passando come parametro un continente, l’invio dei contagi per regioni nel caso solamente dell’italia e l’invio di notizie per continenti.

Molto semplicemente $text è il messaggio che il bot riceve dall’utente e decide che operazione eseguire in base ai case presenti nello switch, grazie a questo il bot sarà in grado di prelevare mediante la connessione al database effettuata nel datamodel i dati sui contagi e le notizie aggiornati e di mostrarli a video grazie alle tre funzioni precedentemente spiegate.

