PROGETTO TO-DO BOT TELEGRAM

Essien Favour – 5f Informatica

INDICE

Realizzazione Del Bot

Realizzazione di un bot Telegram per gestire e tenere traccia dei To-Do(task da fare) degli utenti.

Nome: TodoBot

Username: @todorecorderbot

Comandi:

Commandi normali;

- /start - start.
- /help - per vedere tutti i commandi possibili.
- /registrati - per effettuare una registrazione.
- /login - per accedere.
- /logout - per uscire.
- /cancel - per annulare un processo.

Gestione Todo;

- /view_todo - per vedere tutti i todo.
- /view_todo_fatto - per vedere i todo fatti.
- /view_todo_nonfatto - per i todo non fatti.
- /insert_todo - per inserire un todo
- /update_todo - per aggiornare un todo.
- /update_stato_todo - per aggiornare lo stato di un todo.
- /delete_todo - per eliminare un todo.
- /fatto - per cambiare lo stato un todo a 'fatto'.
- /non_fatto - per cambiare lo stato di un todo a 'non fatto'.

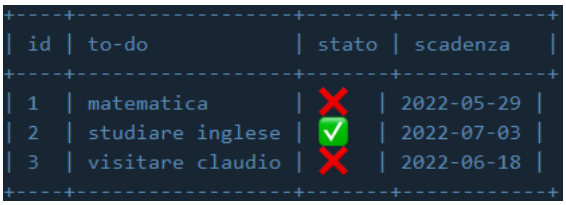
Commando Help

Con il commando /help, il bot controlla se l'utente è loggato: Se si, mostra all'utente i commandi per gestire i propi todo come /view_todo e /update_todo; Se no, mostra all'utente i commandi normali come /login e /logout.

Gestione Degli Utenti

I commandi /registrati e /login consente all'utente di registrarsi e accedere al proprio account per gestire i todo utilizzando un nome utente e una password mentre il commando /logout permette all'utente di disconnettersi dal prorio account.

Visualizzazione dei Todo



id	to-do	stato	scadenza
1	matematica	❌	2022-05-29
2	studiare inglese	✅	2022-07-03
3	visitare claudio	❌	2022-06-18

Con il commando /view_todo, il bot recupera i todo associati all'utente loggato dal database e li invia in forma tabellare all'utente. I todo con lo stato '❌' sono i todo non ancora fatti mentre quelli con lo stato '✅' sono i todo fatti.
Per essere più specifico l'utente può usare i commandi /view_todo_fatto o /view_todo_nonfatto per visualizzare **solo** i todo fatto o **solo** i todo non fatti.

Inserimento di un Todo

Quando l'utente inserisce il commando /insert_todo il bot chiede la descrizione del todo. Dopo che l'utente ha inviato la descrizione del todo, il bot chiede la data di scadenza e di seguito salva il todo nel database come un todo 'non fatto' e manda la lista dei todo all'utente in forma tabellare.

Aggiornamento di un Todo

Per aggiornare un todo viene utilizzato il commando /update_todo. Quando l'utente invia il commando, il bot invia la tabella dei todo all'utente e gli chiede di scegliere l'id del todo da aggiornare. Dopo aver ricevuto l'id dall'utente, il bot chiede la descrizione e la data aggiornata del todo e di seguito, aggiorna il todo nel database e invia la versione aggiornata all'utente in forma tabellare.

Aggiornamento dello stato di un Todo

Per aggiornare lo stato di un todo, viene utilizzato il commando /update_ stato _todo. Quando l'utente invia il commando, il bot invia la tabella dei todo all'utente e gli chiede di scegliere l'id del todo da aggiornare. Dopo aver ricevuto l'id dall'utente, il bot chiede dall'utente di inviare il commando /fatto - se il todo è fatto o /non_fatto - se il todo non è fatto. Di seguito, il bot aggiorna lo stato del todo nel database e invia la versione aggiornata all'utente in forma tabellare.

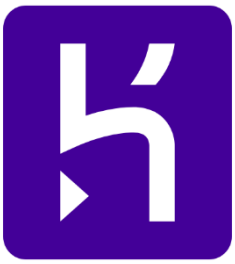
Eliminazione di un todo

Per eliminare un todo viene utilizzato il commando /delete_todo. Quando l'utente invia il commando, il bot invia la tabella dei todo all'utente e gli chiede di scegliere l'id del todo da eliminare. Dopo aver ricevuto l'id dall'utente, il bot elimina l'id nel database e invia all'utente la lista aggiornata dei todo in forma tabellare.

Software Utilizzati

***Telegram**: un servizio di messaggistica istantanea e broadcasting basato su cloud che lavora su desktop e su mobile.*

***Heroku**: una platform as a service (PaaS) sul cloud che supporta diversi linguaggi di programmazione come node.js, python ecc.*



***Node js**: un sistema open source multipiattaforma orientato agli eventi per l'esecuzione di codice JavaScript, costruita sul motore JavaScript V8 di Google Chrome.*

***Npm (Node Package Manager)**: un gestore di pacchetti per il linguaggio di programmazione JavaScript. È il gestore di pacchetti predefinito per l'ambiente di runtime JavaScript Node.js.*

Database

Il database predisposto per questo bot possiede due tabelle:
riportiamo in seguito il modello ER del database e descriverò il modello relazionale più nel dettaglio.

Schema Logica;

Utenti (IDUtente, username, password)

- Tabella per salvare i dati degli utenti.

Todo (IDTodo, IDUtente, todo, stato, scadenza)

- Tabella per salvare i todo degli utenti..

Pachetti e Librerie Utilizzati

Telegraf.js

Telegraf è una libreria che ti semplifica lo sviluppo di bot Telegram utilizzando JavaScript o TypeScript. E' un framework moderno dell'API di Telegram Bot per Node.js.

Installazione

```
$ npm install telegraf --save
```

Telegraf-Calendar-Telegram

Usando questo semplice calendario in linea puoi consentire a un bot di Telegram di chiedere date. Questa libreria è costruita utilizzando Telegraf Framework.

Installazione

```
$ npm i telegraf-calendar-telegram --save
```

PrettyTable

Libreria per la visualizzazione di dati tabulari in un formato ASCII visivamente accattivante.

Installazione

```
$ npm i prettytable --save
```

Express

Un framework web veloce, semplice e minimalista per node.

Installazione

```
$ npm i express
```

Dotenv

E' un modulo a dipendenza zero che carica le variabili di ambiente da un file .env in process.env.

Installazione

```
$ npm i dotenv mysql --save
```

Ejs

E' un semplice framework di creazione di modelli che ti consente di generare markup HTML con JavaScript semplice.

Installazione

```
$ npm i ejs --save
```

Commander

E' un framework a riga di comando leggero, espressivo e potente per node.js.

Installazione
$ npm i commander –save

Riga di commando

E' stato usato il framework commander per svillupare un interfaccia di riga commando che permette di visualizzare i todo di tutti gli utenti.

Commandi:

- node index.js -v **–** per vedere i todo di tutti gli utenti.
- node index.js -v -n "nome-utente" **-** per vedere i todo dell'utente specificato.

Sito Web

E' stato usato il framework ejs per svillupare un sito web che permette agli utenti di vedere i propri todo.

Url: todobot-telegram

Hosting

Database:

Il database è stato hostato sul sito remotemysql.com

Processo:

Prima di tutto, vai alla pagina di accesso e registrati per un nuovo account. Una volta fatto, vai alla dashboard e crea un nuovo database MySQL: il sito ti fornirà quindi un nome utente, un nome del database e una password. Il prossimo passo è andare alla

pagina myphpadmin e accedere con le tue credenziali. Usando myphpadmin, puoi gestire il tuo database e usarlo.

Bot:

Il bot è stato hostato su Heroku.

Processo:

Vai a Heroku Dev Center e git install segui le istruzioni per scaricare e installare Heroku CLI e git.

Innanzitutto, hai bisogno di un account Heroku e può essere creato dalla pagina di registrazione.

Dopo avere un account, apri il prompt dei comandi ed esegui il comando:

$ heroku login

Ci chiederà di inserire il nostro account Heroku (e-mail e password) in una finestra del browser. Quindi riceveremo*:*

heroku: Press any key to open up the browser to login or q to exit:
Opening browser to https://cli-auth.heroku.com/auth/browser/xxx
Logging in... done
Logged in as utente@gmail.com

Punta il prompt dei comandi nella directory principale del tuo progetto, quindi crea il repository Git:

$ cd "tuo-progetto"
$ git init
$ git add .
$ git commit -m "initial commit"

Crea l'app Heroku con il comando:

$ heroku create "progetto-application"

Puoi facilmente distribuire la tua app Node.js su Heroku inviando il codice al repository remoto che abbiamo creato nel passaggio precedente. Heroku rileverà automaticamente che si tratta di un'app Node.js e la costruirà di conseguenza:

$ git push heroku master

Per aggiornare il codice al repository remoto di Heroku:

$ git add .
$ git commit
$ git push heroku master

Da Notare:

I contenitori usati a Heroku sono chiamati "**dynos**". I Dynos sono contenitori Linux isolati e virtualizzati progettati per eseguire codice in base a un comando specificato dall'utente.
Se utilizzi un web dyno(gratuita), il tuo bot si spegnerà dopo 30 minuti di inattività, per riaccendere il bot, basta andare sul sito della tua applicazione di Heroku.
Invece, per far funzionare il bot "24 ore su 24, 7 giorni su 7" dovresti pagare per il dyno worker.

Sitografia

- Documentazione npm - https://www.npmjs.com/
- Per varie definizioni - https://it.wikipedia.org
- https://w3schools.com/
- Per installazione e setup di Heroku - https://www.bezkoder.com/deploy-node-js-app-heroku-cleardb-mysql/
- Per la risoluzione di errori nel codice - StackOverflow