Exercise 7-1  Querying crates.io

Develop a program that allows to retrieve meta-data for crates. Start with the simple URL retriever from the lecture and extend it by little until it offers the required features. You can rely on the standard library and the crates \texttt{hyper, hyper-tls, serde, and serde-json}.

a) Modify the URL retriever from the lecture such that it allows TLS-secured HTTPS requests. The crate \texttt{hyper-tls} provides an example that outlines the required changes.

The function should have the following signature:
\begin{verbatim}
fn fetch_crateinfo(url: Uri) -> impl Future<Item=String, Error=()>. As an additional requirement the function should not use \texttt{unwrap} in an unsafe way. Instead, use \texttt{map_err} in order to map error values to the required future error values.
\end{verbatim}

b) Given is the function signature \texttt{fn keywords_from_response(resp: String) \to Option<String>}. Leave this function unimplemented for now but use it to create a new function \texttt{fn crate_keywords(c: &str) \to impl Future<Item=String, Error=()}> that takes a reference to a string (a crate name) and returns a future \texttt{String} representing keywords for that crate.

The keywords for a crate can be gathered by sending a request to the URL \texttt{https://crates.io/api/v1/crates/CRATENAME} and parsing the returned JSON information. For example, the request to \texttt{https://crates.io/api/v1/crates/hyper} yields a JSON document from which keywords can be extracted using the aforementioned function.

c) Write \texttt{fn main():} It shall run \texttt{fn crate_keywords}, print the result to screen and modify the returned future such that it can be used as a task in the call to \texttt{hyper::rt::run}.

d) Write the function \texttt{fn keywords_from_response(resp: String) \to Option<String>} that

\begin{itemize}
  \item deserializes a HTTP response (given as a \texttt{String}) into its JSON representation,
  \item extracts crate keywords from the JSON representation, and
  \item concatenates them into a single \texttt{String}.
\end{itemize}

The keywords can be gathered from the JSON representation with a query like \texttt{json["crate"]}["keywords"] which yields the keywords as a vector of JSON \texttt{Value} data type.

Hints:

\begin{itemize}
  \item Use method \texttt{collect::<Option<Vec<_>>>} to convert from an iterator over \texttt{Option<&str}> into \texttt{Option<Vec<&str>>>}.
  \item Use method \texttt{fold} to concatenate a \texttt{Vec<&str>} to yield a \texttt{String}.
  \item Documentation on Serde and a helpful example can be found at \url{https://docs.serde.rs/serde_json/#operating-on-untyped-json-values}
  \item Use an online JSON formatter like \url{https://jsonformatter.curiousconcept.com/} to get acquainted with the HTTP response body.
\end{itemize}