**Bible Verse Application**

Kacey Morris

Grand Canyon University

Enterprise Applications Programming – CST 247

Professor Sluiter

April 18, 2021

**Technical Approach**

For the Bible Verse Application, I followed the directions very directly. During class time, the entry form for entering Bible verses into the database was taken out of the requirements, so I did not implement that portion of the directions. In the next section, the directions specify a form input with several form elements. I understood this request as a single form with all inputs required to search for a verse in the Bible. Data validation to check for data types and values are implemented for each field, an error message displaying if the user does not provide a selection for each field. All pages of the application share a common header and footer and, within the header, a navigation bar is available to access the search form from any page. Because the ASP.NET Core framework is used, as well as bootstrap features, the entire site is responsive. The header also includes the Logo and Title of the site. When a user uses the search form to search for a verse, the user is taken to a page to display the verse if it was found, or, if not found, a page informing the user of this fact. Throughout the application, I utilize N-Layer architecture with the use of Views, Business, and Data Layers, and Object-Oriented principles with the use of Models. Dependency Injections are also implemented through constructor parameters, and a logger has been created to log important information about the method interactions. Finally, comments throughout the code explain the purpose and flow of the coding process.

Classes:

* Bible Controller – connects interactions with the view to business service methods and actions.
* Search Term – defines the properties and methods of the form inputs included in the search view.
* Verse Details – defines the properties and methods of a single verse.
* Bible Business Service – accesses the data service while enforcing N-Layer design and integrity.
* Bible DAO – access the database to retrieve Bible verse information.
* My Logger – logs information when utilized to keep track of interactions in a log file.

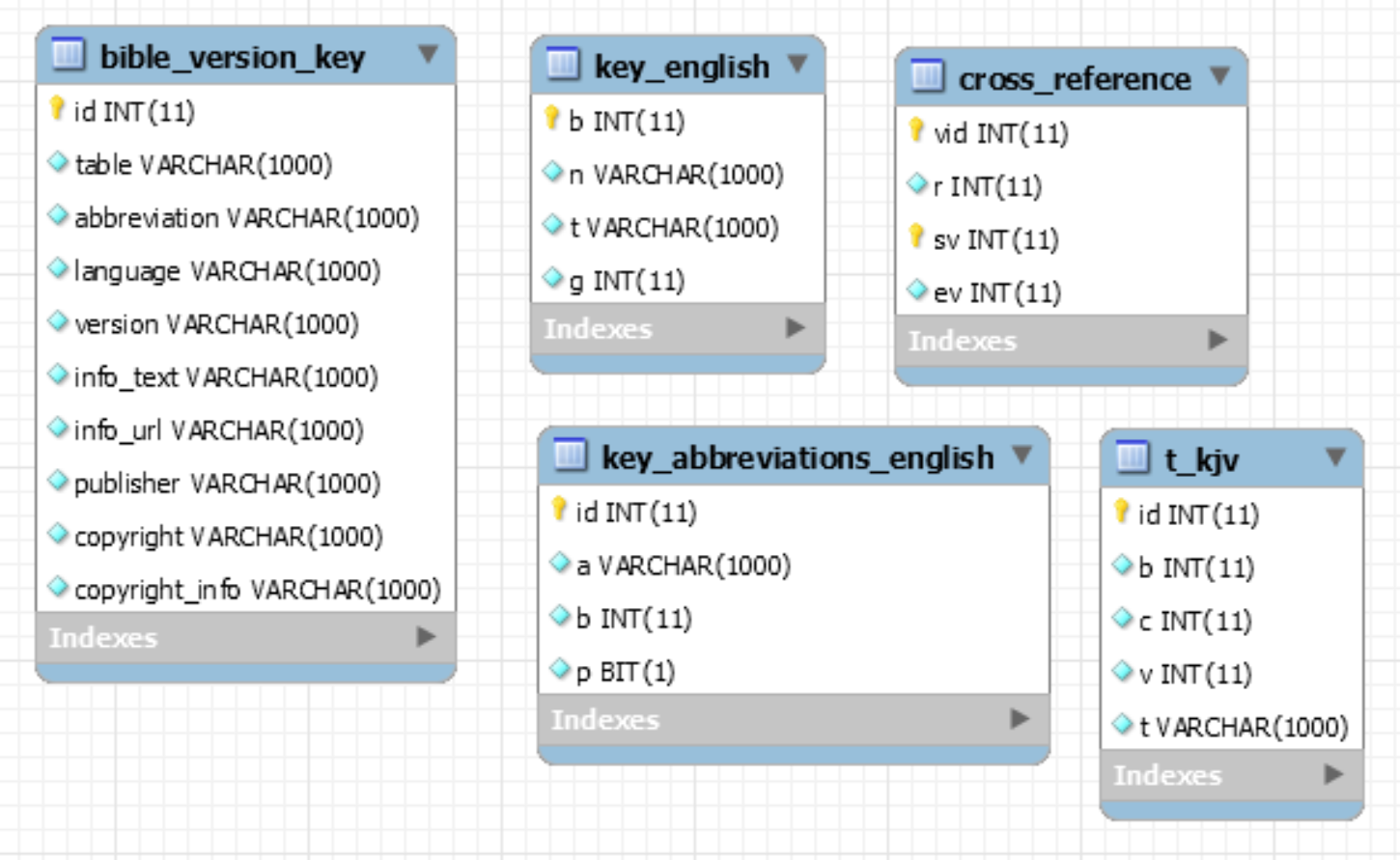
Methods:

* BibleController.Search() – returns the IActionView for the search form
* BibleController.Result(SearchTerm) – utilizes the business service and returns either the results page or the results not found page
* BibleBusinessService.FindVerse(SearchTerm) – utilizes the data service to return a Bible verse object
* BibleDAO.FindVerse(SearchTerm) – accesses the database to find a Bible verse based on the form input
* MyLogger.Debug(), Error(), Info(), Warning() – basic logging methods

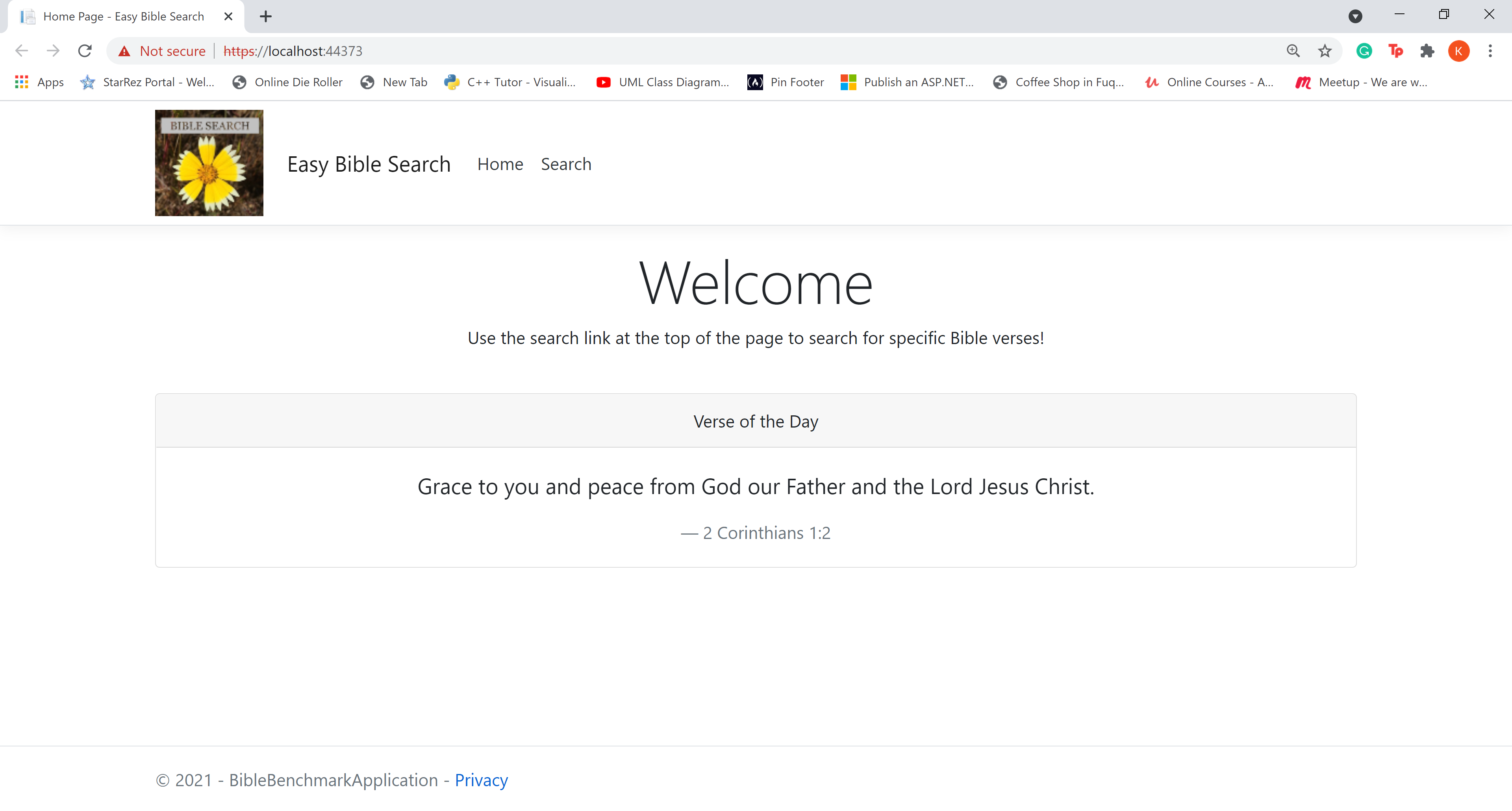
Variables:

* IBibleBusinessService service – uses dependency injections, business service for controller
* IBibleDAO dao – uses dependency injections, data access object for business service
* String connectionString – saves the connection string for the database, used when accessing the database
* VerseDetails foundVerse – the verse found when searching the database using the search term
* String sqlStatement – the SQL statement used to access the database
* SearchTerm term – the object made from the input from the user about which verse to search

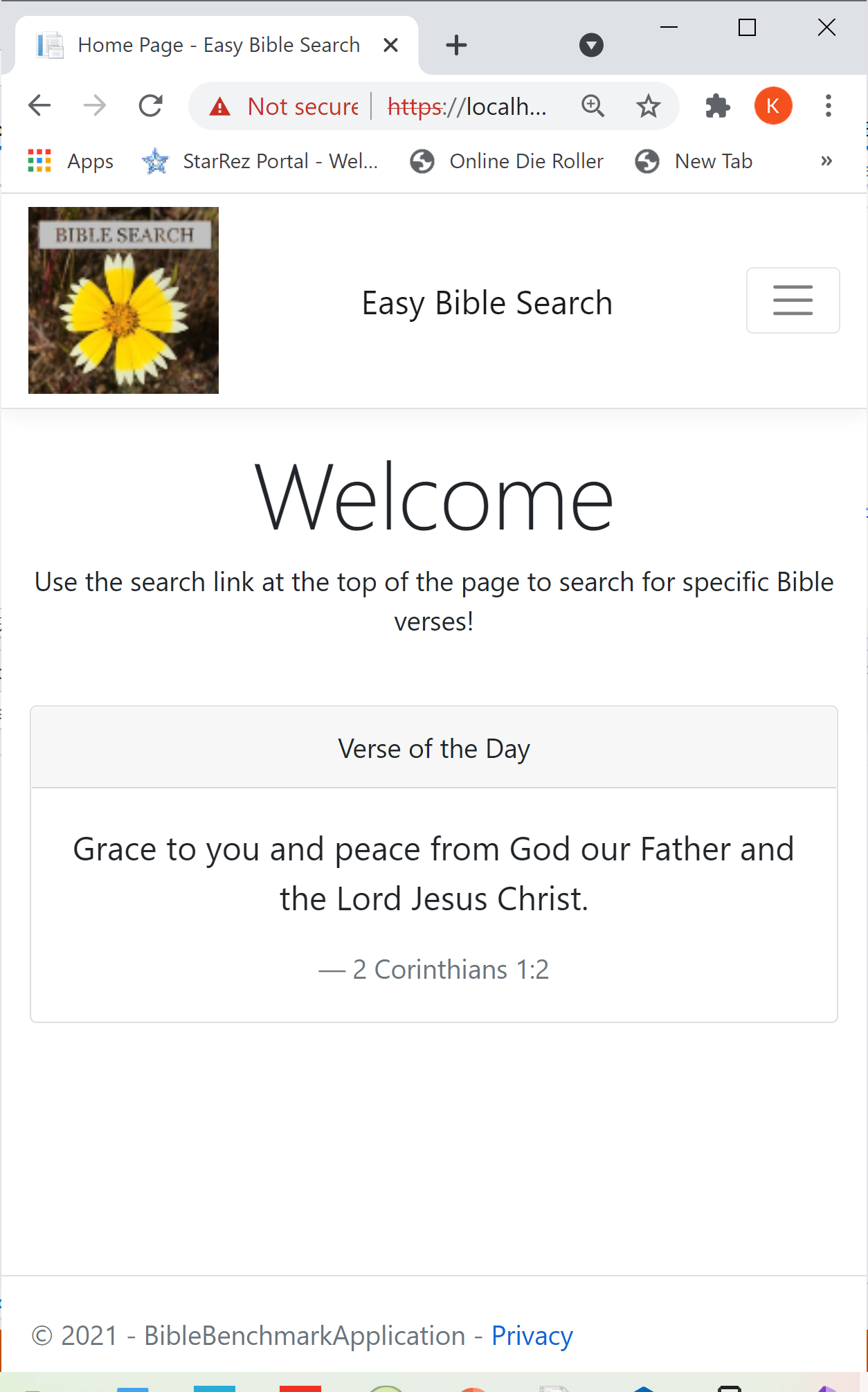
**ER Diagram**



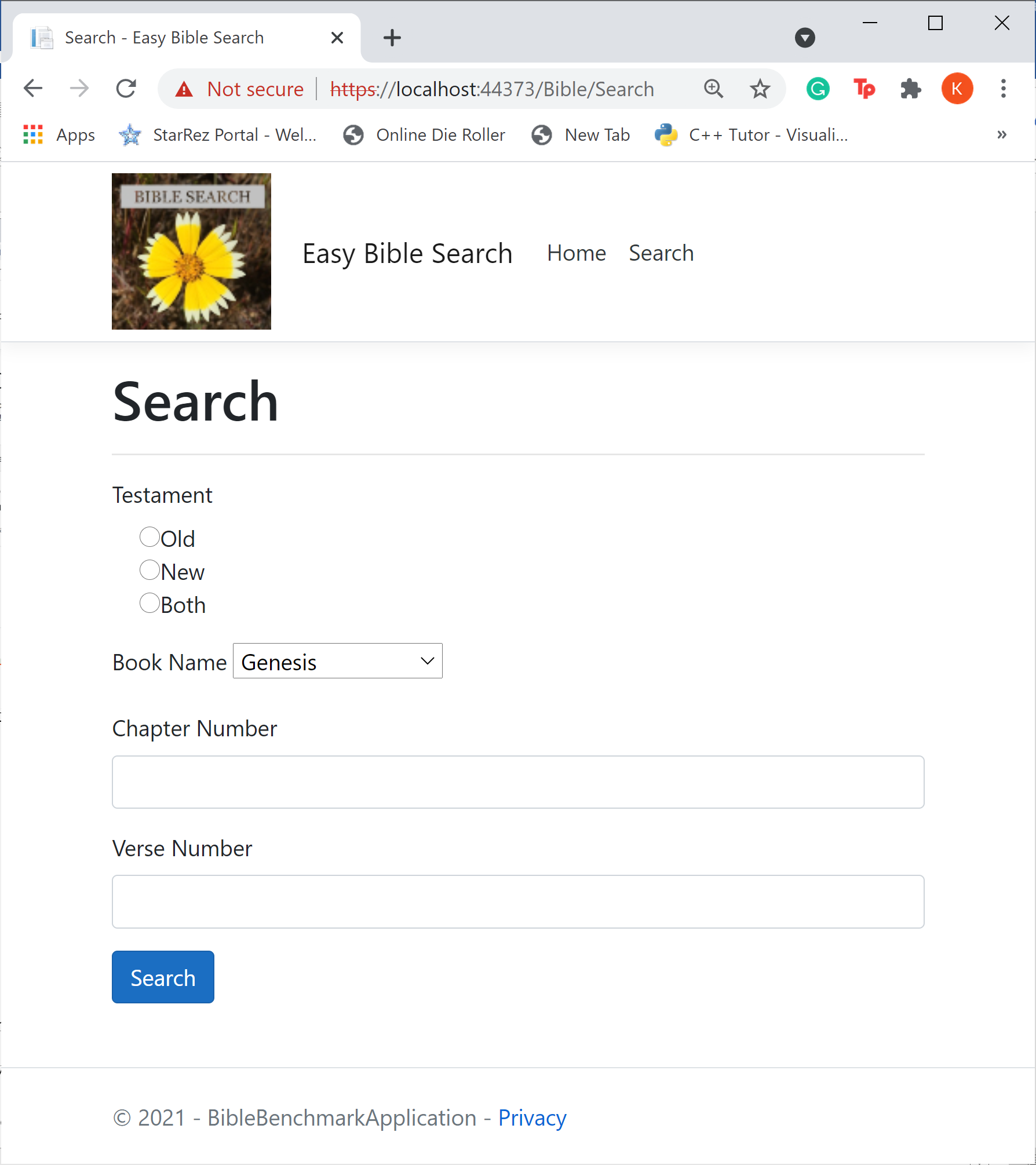
**Screenshots**



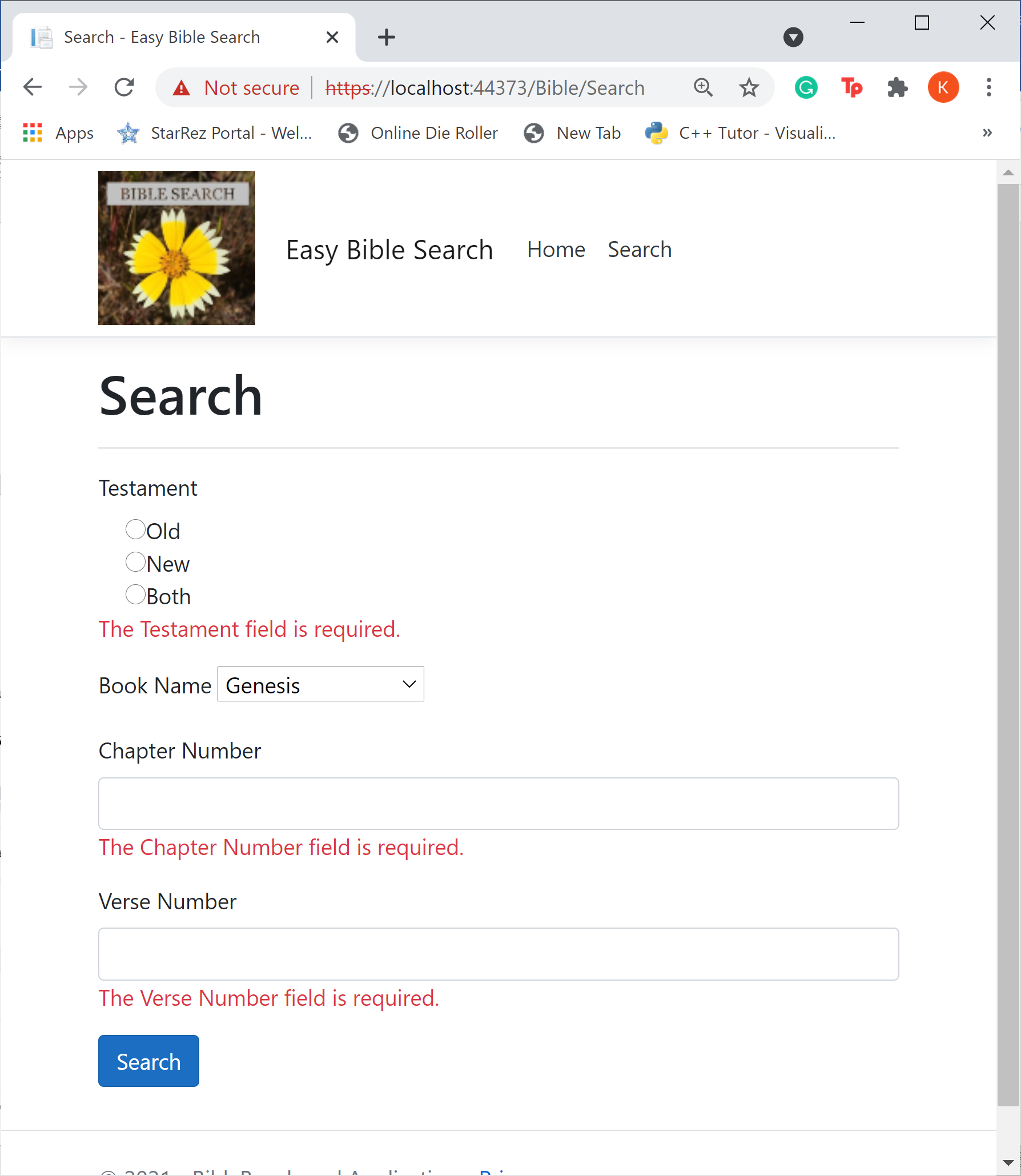
The home screen which shows a common header, including the Logo and Title, and footer, a link that is always accessible in the nav bar for search, and a setup for future implementation of the Verse of the Day.



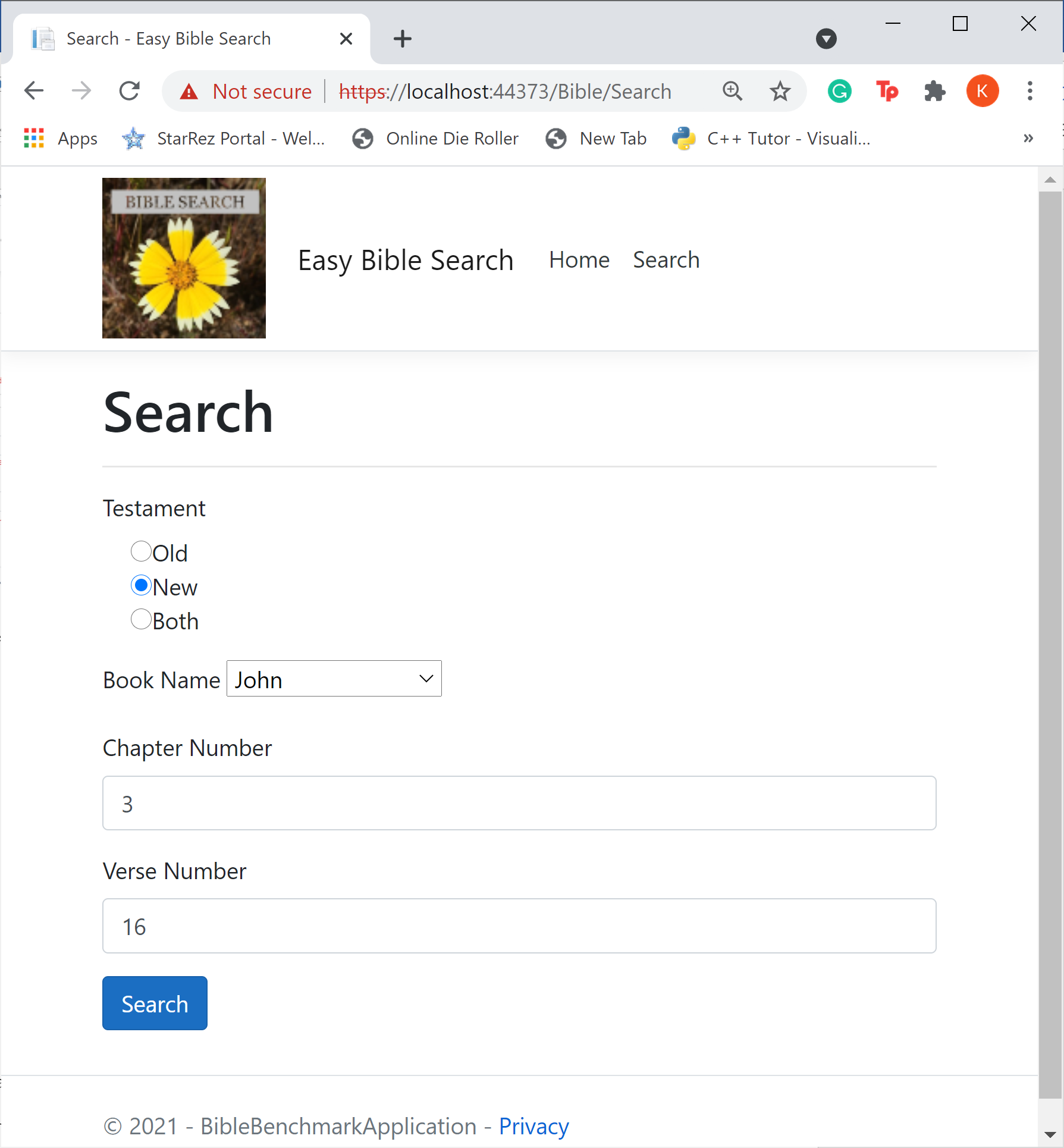
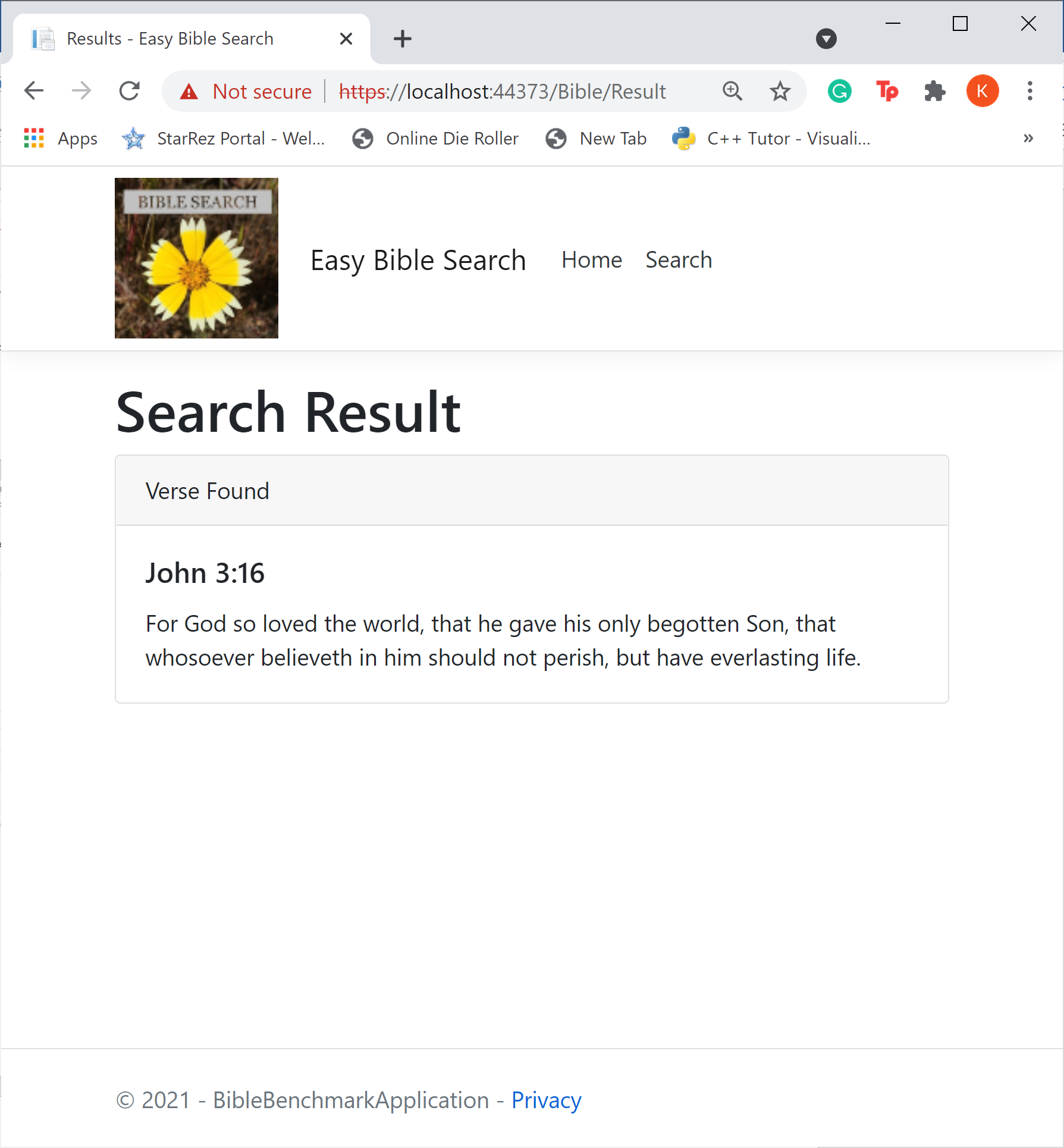
Responsive page design adapting to a change in page size.



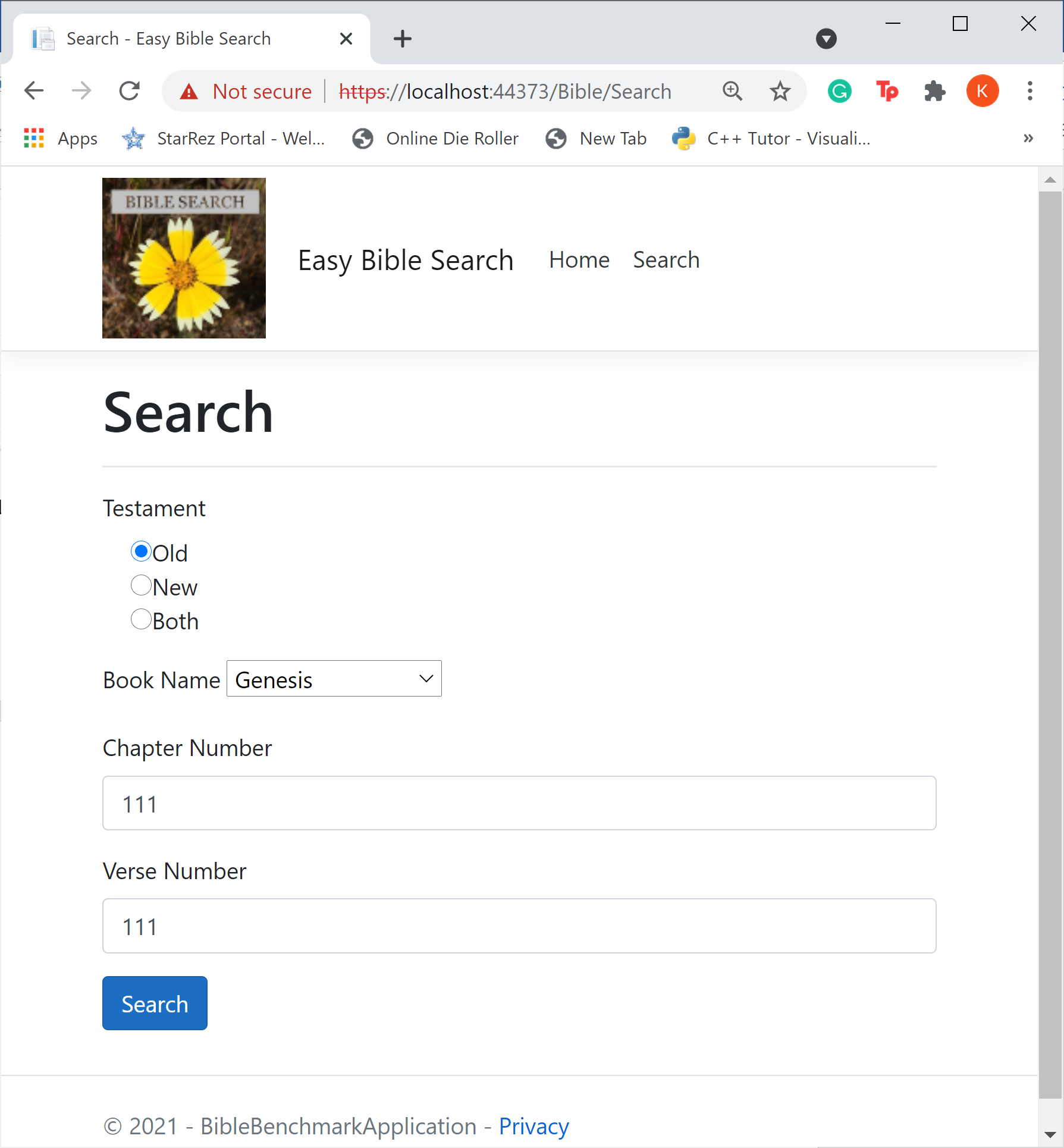
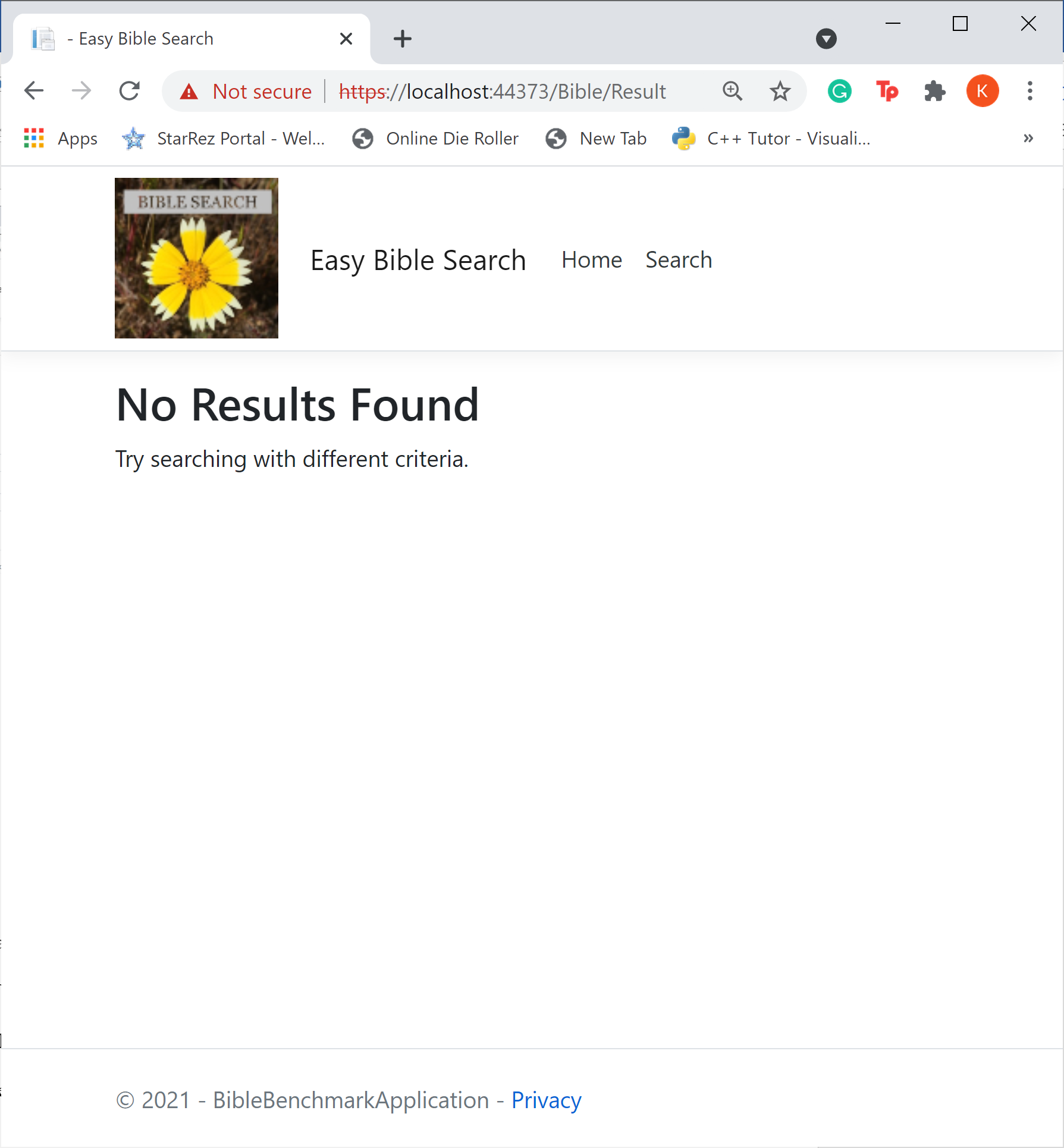
The search form with fields for all required search criteria.



Data validation errors requiring data entry for form submission.

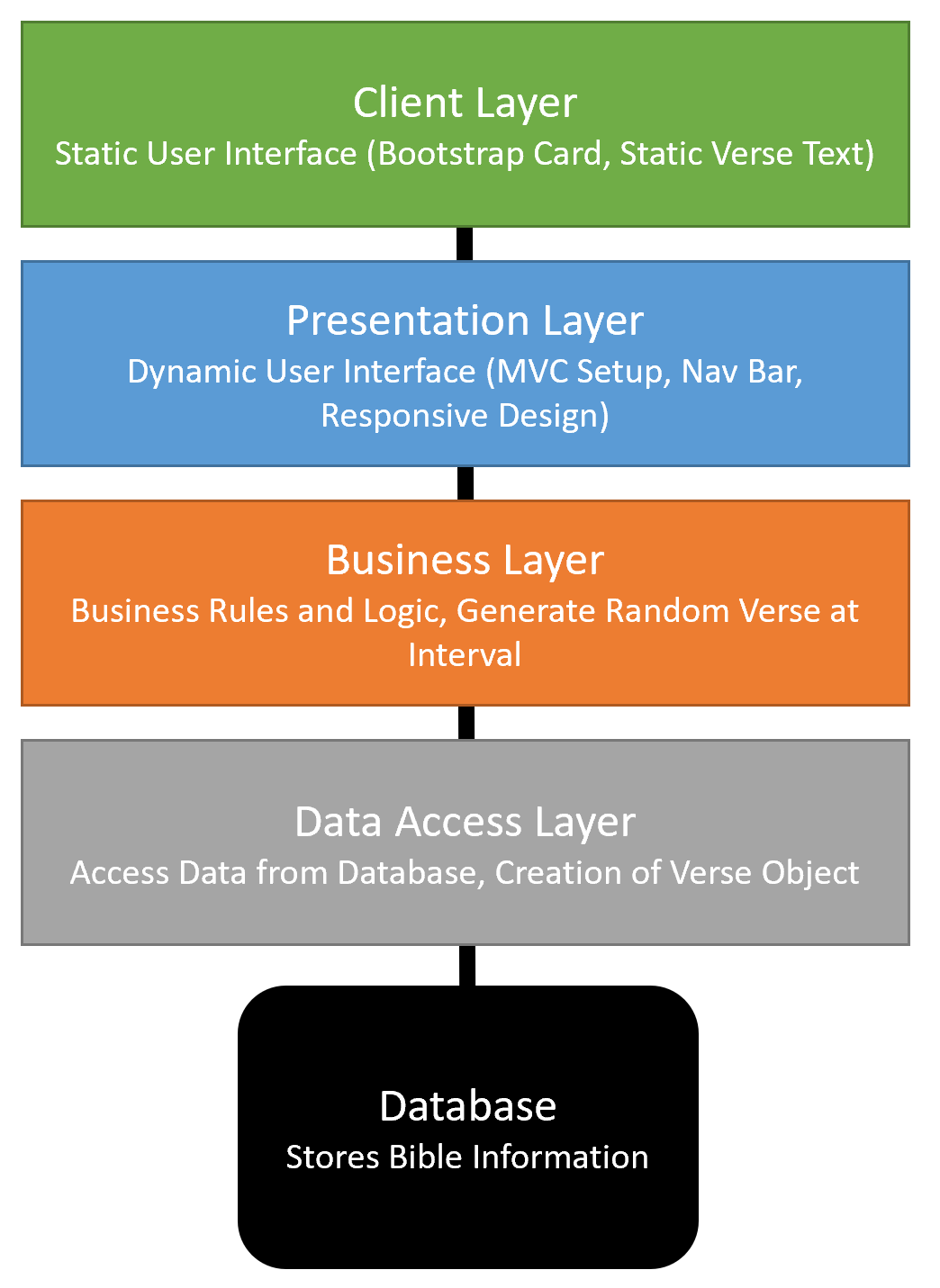
Search fields and search results when specifying John 3:16.

When a verse that does not exist is searched, a failure page prompts the user to try searching for a different verse.

**Verse of the Day**

A Bible Verse of the Day feature could be easily implemented within this application. In my application, I implemented a layout that could be used for this feature, where the verse is displayed in a card format on the main page of the application. To implement a different verse every day, a daily verse generator would need to be constructed in the back-end code. Personally, I would use a random number generator to choose a random book, chapter, and verse number each day to then be passed to the home controller index page and displayed in the card for the entirety of the day. This generation could use the same business and data service already implemented in the application, just with a different action result in the controller. A Timer could be used to generate a new random verse at the same time every day. Although the “Verse of the Day” section is currently on the home page, another page could be created specifically for daily verses and maybe a list of previous verses for the current month could be displayed there as well. A Restful API could also be generated so the daily bible verse with all necessary properties could be accessed outside the application.



**Common Good**

The Christian Worldview influences all aspects of life and programming is not an exception. One of the beliefs commonly held in the Christian Worldview is to contribute to the common good and assist others whenever possible. By developing knowledge about enterprise-programming tools, business problems can be solved to improve the common good. When a programmer with Christian Worldview creates an application, the enterprise tools developed can be applied to increase the integrity of data to keep the general public safer online. This drive to protect the common good is motivated by the prioritization of not only what is good for others, but what would be considered good in God’s eyes. In the creation of the Bible verse application, public access to the Bible is made possible, along with an easy search that ensures the user is receiving the information desired. The responsive design of the site also allows for access to the Bible at any time if the user has a mobile device. This design is also motivated by the Christian Worldview by adding features with the expectation of improving the lives of those using the application.

**GIT Repository**

GitHub Link: <https://github.com/KMorris63/BibleBenchmark.git>

**LOOM Video(?)**

Video Link: <https://www.loom.com/share/c2cc4d36fec74ff29e3fc68a0d64317f>