**Design document – To-Do Application**

**Overview**

This document is the complete analysis of the **web-based** To-Do application which will cover everything from the **purpose** of this application, **Tech stack** to see what are the technical requirements, **Data design** to see the flow of data across the application**,** and last but not the least working **user interface** of the application to see the final product.

**Purpose**

There are times when we forget our day-to-day activities very often, for that we need an easy-to-access application in which we can list down our activities that need to be done in that way this application can help to keep a **track** of our **day-to-day activities**.

**Tech stack**

To build an organized/sorted and configurable application we are following the **MVC architecture**

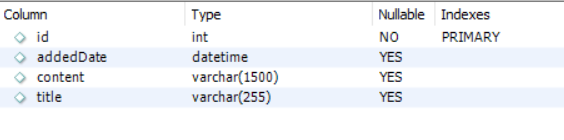
* **JSP:** Java Server Pages are used to create the **View** of the application i.e. the user interface from which the user can interact.
* **Servlets:** Another java web-based tech that will help in **Controlling** the application i.e. performing all the backend tasks done via Servlets
* **Hibernate:** To handle the **Model/Data** view of the application we are using Hibernate so that our focus will be less on SQL queries and more on development.

**Functional Requirements**

There are some key requirements to fulfill the purpose of this application:

* **“Add” functionality:** To add new notes, we should have the form in the front end for adding *new notes*.
* **“Delete” functionality:** We should have the delete option along with each note to delete unnecessary notes.
* **“Update” functionality:** Sometimes we might need to update the previous notes, for that along with each note we should have an *update option* as well.
* **“Show all notes” functionality:** This is important because by looking at all the notes only we can check the tasks we need to perform.

**Data Design**

****

Above is the schema of the **“Notes”** table:

* **ID column:** From a coding standpoint, we won’t be dealing with this field but this will surely be one of the keys while **retrieving** the records from the database.
* **Added-Date column:** This column will hold the current **time-stamp** of when the notes are *added or updated*
* **Content column:** This field will hold the main notes content – keep a note on the **Type** and word limit too because if we will enter more than 1500 words then the application will throw an error.
* **Title column:** This field will hold the title of the Note i.e. **head** of the content.

**Data Flow**

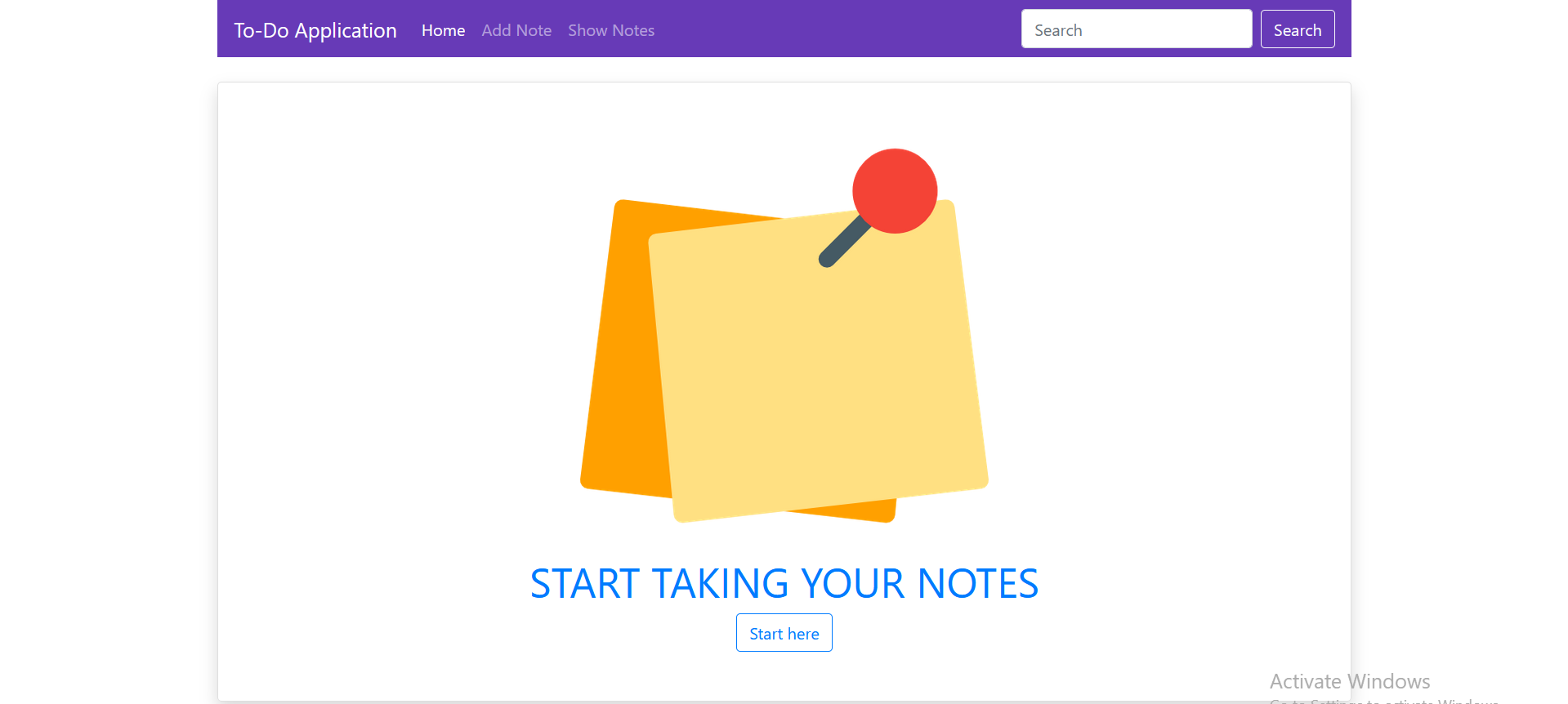
****

**User Interface**

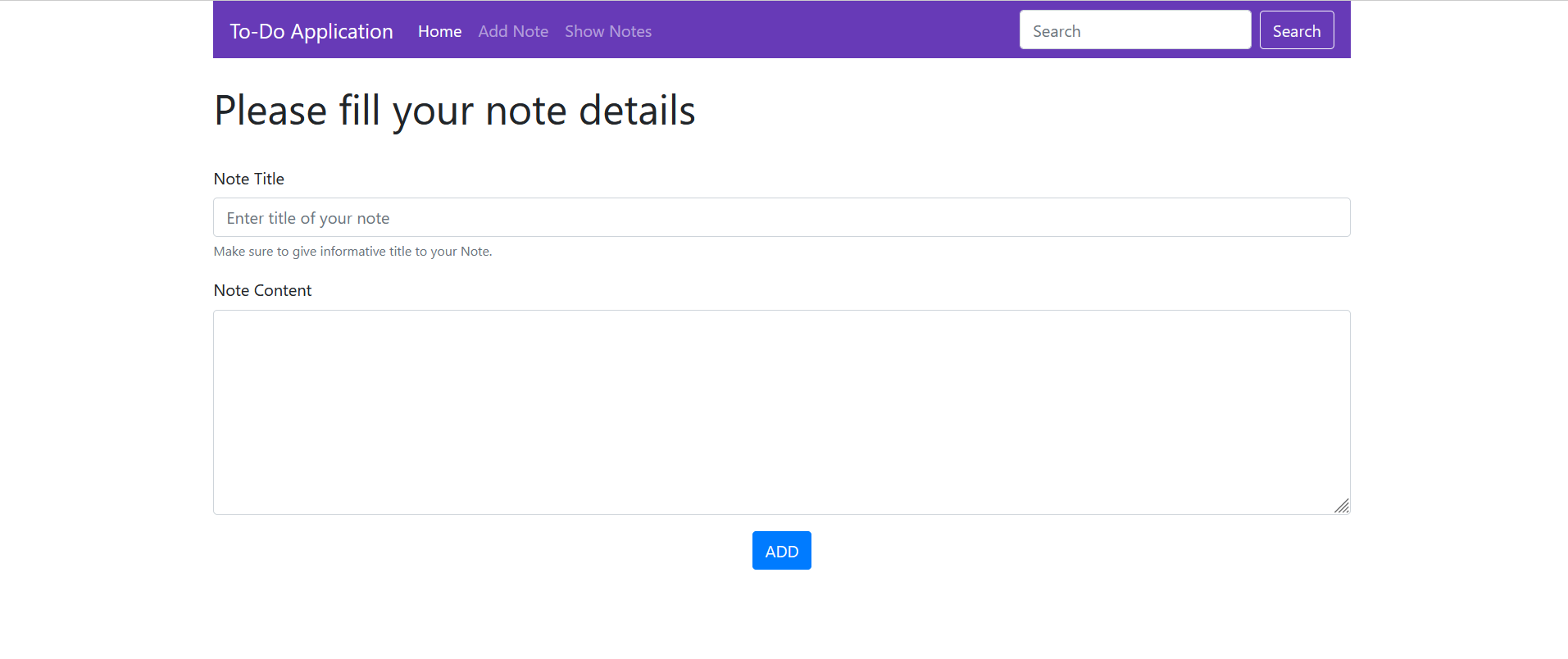
**Home page:**

**Navigation –**

* **Add Note:** For adding new notes in the database
* **Show Note:** Will show all the notes in the form of cards
* **Home:** This will bring you back to this page – click on **START HERE** to Add a new Note

****

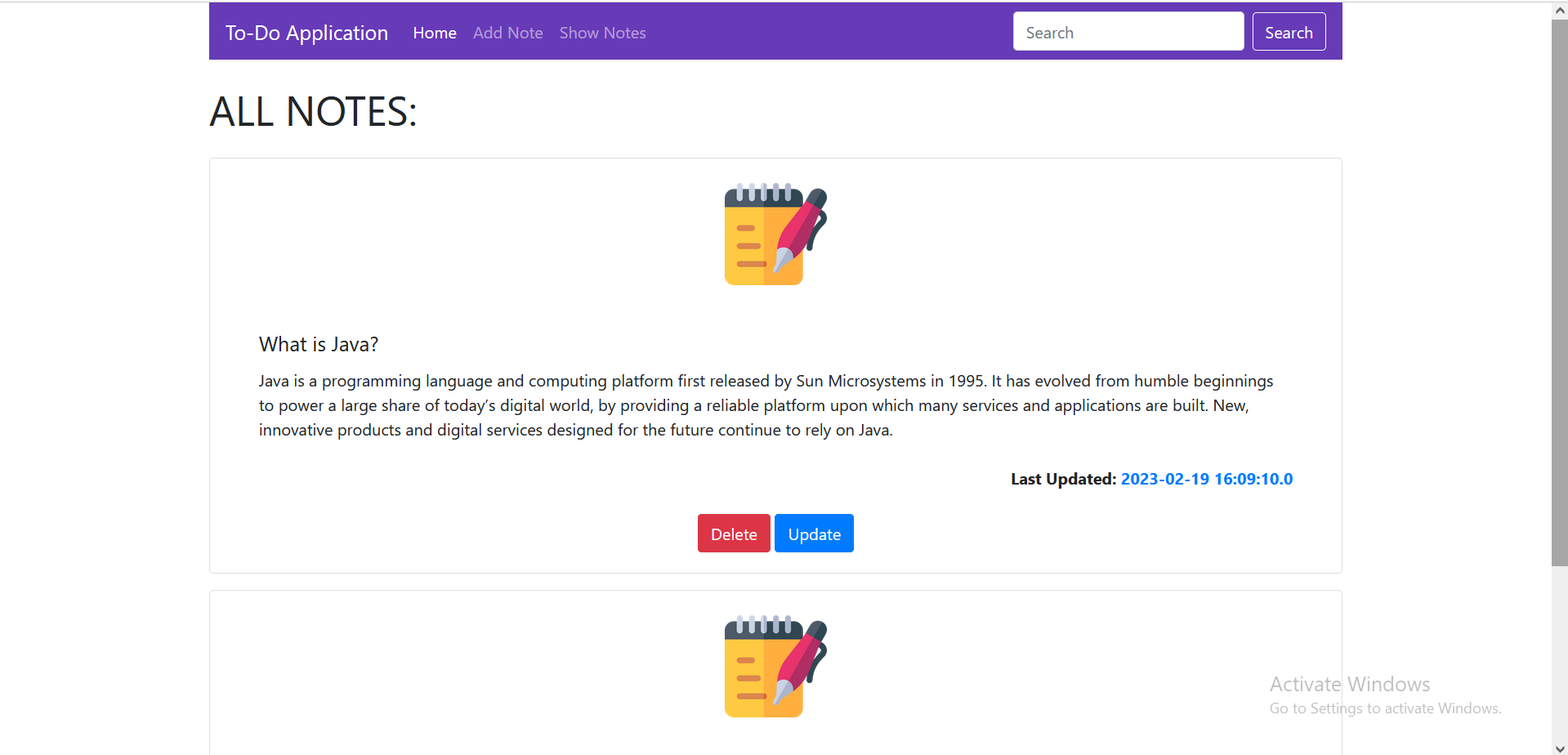
**Add New Note: Fields:** Note Title and Note Content

****

**Show All Notes:**

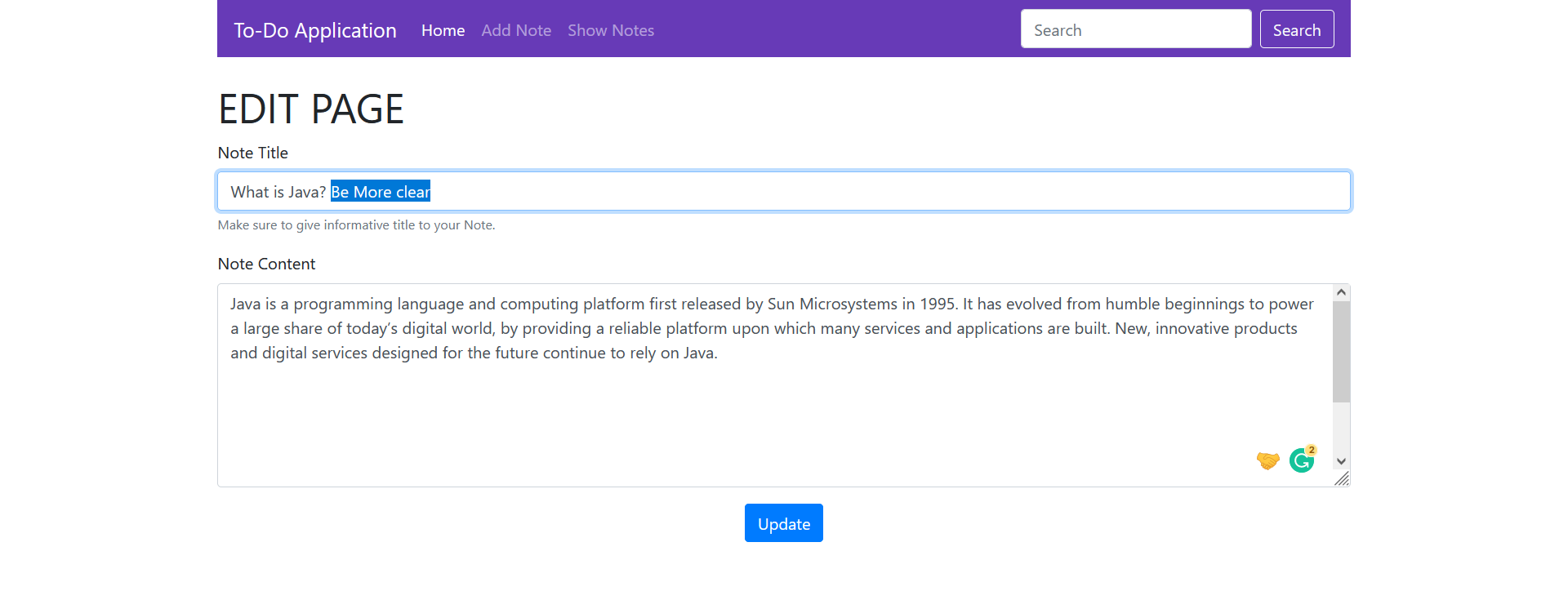
Corresponding to each **Note card** we have the option,

* To **Delete**
* To **Update**
* Current **timestamp**

****

**Updating Notes:**

The Note that we opt to **edit**, will be populated in this form so that we can make considerable changes above the existing Note **Title** and Note **content**

****