**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **03/06/2020** | **Name:** | **Shilpa.C** |
| **Course:** | **Digital design using HDL** | **USN:** | **4al17ec086** |
| **Topic:** | 1. EDA Playground Tutorial Demo Video 2. How to Download And Install Xilinx Vivado Design Suite 3. Vivado Design Suite for implementation of HDL code | **Semester & Section:** | **6th , B sec** |
| **Github Repository:** | **shilpa-c** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session** |
| **Report – Report can be typed or hand written for up to two pages.**  **INTRODUCTION**   1. [Log in](http://eda-playground.readthedocs.io/en/latest/login.html). Click the **Log in** button (top right) Then either  * click on Google or Facebook or * register by clicking on ‘Register for a full account’ (which enables all the simulators on EDA Playground)  1. Select your language from the **Testbench + Design** menu. 2. Select your simulator from the **Tools & Simulators** menu. Using certain simulators will require you to supply [additional identifcation information](http://eda-playground.readthedocs.io/en/latest/login.html). 3. Type in your code in the **testbench** and **design** windows. 4. Click **Run**.   What is EDA Playground?  EDA Playground gives engineers immediate hands-on exposure to simulating SystemVerilog, Verilog, VHDL, C++/SystemC, and other HDLs. All you need is a web browser. The goal is to accelerate learning of design/testbench development with easier code sharing and simpler access to EDA tools and libraries. Tools & Simulators For settings and options documentation, see [Tools & Simulators Options](https://eda-playground.readthedocs.io/en/latest/settings.html#tools-simulators-options-label)  Available tools and simulators are below. EDA Playground can support many different tools. [Contact us](http://www.doulos.com/) to add your EDA tool to EDA Playground. Simulators  * [Synopsys VCS](http://www.synopsys.com/Tools/Verification/FunctionalVerification/Pages/VCS.aspx)   + Commercial simulator for VHDL and SystemVerilog * [Cadence Incisive](http://www.cadence.com/products/fv/enterprise_simulator/pages/default.aspx)   + Commercial simulator for VHDL and SystemVerilog (VHDL simulation not yet implemented on EDA Playground) * [Aldec Riviera-PRO](https://www.aldec.com/en/products/functional_verification/riviera-pro)   + Commercial simulator for VHDL and SystemVerilog   + [Riviera-PRO Product Manual](https://www.aldec.com/en/support/resources/documentation/manuals) (registration required) * [Incisive Specman Elite](http://www.cadence.com/products/fv/enterprise_specman_elite/pages/default.aspx)   + Commercial simulator that supports [e Verification Language, IEEE 1647](http://www.cadence.com/products/fv/pages/e_overview.aspx)   + Works with [Cadence Incisive](http://www.cadence.com/products/fv/enterprise_simulator/pages/default.aspx)   + [Hello e World Video Tutorial](https://www.youtube.com/watch?v=A07FJF0RvH0) * [GHDL](http://ghdl.free.fr/)   + an open-source simulator for the VHDL language   + fully supports the 1987, 1993, 2002 versions of the IEEE 1076 VHDL standard and partially the latest 2008 revision (well enough to support fixed\_generic\_pkg or float\_generic\_pkg) * [Icarus Verilog](http://iverilog.icarus.com/)   + Version 0.10.0 (devel) supports several SystemVerilog features. * [GPL Cver](http://sourceforge.net/projects/gplcver/) * [VeriWell](http://sourceforge.net/projects/veriwell/)  Compilers and Interpreters  * [C++](http://gcc.gnu.org/) * [Perl](http://www.perl.org/) * [Python](http://www.python.org/) * [Csh (C Shell)](http://en.wikipedia.org/wiki/C_shell)  Synthesis Tools NOTE: The synthesis tools will only process code in the right Design pane. The code in the left Testbench pane will be ignored.   * [Yosys](http://www.clifford.at/yosys/)   + [Yosys on GitHub](https://github.com/cliffordwolf/yosys) * [The Verilog-to-Routing (VTR) Project](http://code.google.com/p/vtr-verilog-to-routing/)  Frameworks For settings and options documentation, see [Languages & Libraries Options](https://eda-playground.readthedocs.io/en/latest/settings.html#languages-libraries-options-label)  Available frameworks: SystemVerilog and Verilog  * [Doulos \*Easier UVM\*](http://www.doulos.com/knowhow/sysverilog/uvm/easier_uvm_generator/) * [SVUnit](http://www.agilesoc.com/open-source-projects/svunit/) - unit testing framework for Verilog/SystemVerilog modules, classes, etc.   + [SVUnit on SourceForge](http://sourceforge.net/projects/svunit/) * [TL-Verilog](http://www.redwoodeda.com/) - extends SystemVerilog with new language constructs for pipelines and transactions  Python  * [MyHDL](http://www.myhdl.org/) - a Python based hardware description language (HDL)   + [MyHDL Manual](http://www.myhdl.org/doc/current/)   + [MyHDL on Bitbucket](https://bitbucket.org/jandecaluwe/myhdl) * [Migen](https://migen.readthedocs.org/en/latest) - a Python toolbox for building complex digital hardware   + [Migen on GitHub](https://github.com/m-labs/migen)   + [Migen from M-Labs](http://milkymist.org/3/migen.html) * [cocotb](http://cocotb.readthedocs.org/en/latest/index.html) - a coroutine based cosimulation library for writing VHDL and Verilog testbenches in Python   + [cocotb on GitHub](https://github.com/potentialventures/cocotb)  Libraries & Methodologies For settings and options documentation, see [Languages & Libraries Options](https://eda-playground.readthedocs.io/en/latest/settings.html#languages-libraries-options-label)  Available libraries and methodologies: SystemVerilog and Verilog  * [UVM - Universal Verification Methodology](http://www.accellera.org/downloads/standards/uvm)   + [UVM 1.2 Class Reference](https://eda-playground.readthedocs.io/en/latest/_static/uvm-1.2/index.html)     - [What’s New in UVM 1.2](http://www.youtube.com/watch?v=V2l4lBlsh7k&list=SPScWdLzHpkAdYPk_jgxRgOPisTm3-7U6A) on YouTube   + [UVM 1.1d Class Reference](https://verificationacademy.com/verification-methodology-reference/uvm/docs_1.1d/html/) * [OVM - Open Verification Methodology](https://verificationacademy.com/topics/verification-methodology)   + [OVM 2.1.2 Class Reference](https://verificationacademy.com/verification-methodology-reference/ovmworld/docs_2.1.2/html/index.html)   + [OVM 2.1.2 User Guide](http://www.specman-verification.com/source_bank/ovm-2.1.2/ovm-2.1.2/OVM_UserGuide.pdf) * OVL - Open Verification Library   + [**OVL Library Reference Manual**](https://eda-playground.readthedocs.io/en/latest/_downloads/96ac75b4dcf1d07157888c251809f0c0/ovl_lrm.pdf)   + [**OVL Quick Reference**](https://eda-playground.readthedocs.io/en/latest/_downloads/34a76a1c8e4d29fde51df6a3b0f69ed5/ovl_quick_ref.pdf) * [ClueLib](https://github.com/cluelogic/cluelib) - A generic class library in SystemVerilog   + [ClueLib API Documentation](http://cluelogic.com/tools/cluelib/api/framed_html/index.html) * [svlib](http://www.verilab.com/resources/svlib/) - A Programmer’s Utility Library for SystemVerilog   + [**svlib User Guide**](https://eda-playground.readthedocs.io/en/latest/_downloads/78dee4a6e5661e1da9ffb9a3b982d48c/svlib-userguide-0.3.pdf)  VHDL  * OVL - Open Verification Library   + [**OVL Library Reference Manual**](https://eda-playground.readthedocs.io/en/latest/_downloads/96ac75b4dcf1d07157888c251809f0c0/ovl_lrm.pdf)   + [**OVL Quick Reference**](https://eda-playground.readthedocs.io/en/latest/_downloads/34a76a1c8e4d29fde51df6a3b0f69ed5/ovl_quick_ref.pdf) * PSL - Property Specification Language   + Natively supported by Riviera-PRO * [OSVVM](http://osvvm.org/) - Open Source VHDL Verification Methodology * [UVVM](https://bitvis.no/dev-tools/uvvm/) - Universal VHDL Verification Methodology  C++  * [SystemC](http://www.accellera.org/downloads/standards/systemc) - system level design and simulation in C++   + [SystemC 2.3.1 Class Reference](https://eda-playground.readthedocs.io/en/latest/_static/systemc-2.3.1/sysc/classes.html)   + [TLM 2.0 Class Reference](https://eda-playground.readthedocs.io/en/latest/_static/systemc-2.3.1/tlm/classes.html)   **How to install Xilinx vivado**  Vivado is the software that Xilinx has available for all of its (and Digilent’s) [current FPGAs](http://store.digilentinc.com/fpga-programmable-logic/), so we’ll go through how to download the free WebPACK version of Vivado.  First, we have to download Vivado (or at least the web installer for it) from Xilinx’s website, conveniently at <http://www.xilinx.com/support/download.html>.  Once you get to the download page, choose the appropriate installer for your system; I’m on a Windows 10 machine and don’t feel the need to get a universal, all OS installer, so I’ll choose the Windows Self Extracting Web Installer.  Xilinx likes to know what demographic of people use their software so you’ll need to either sign in to your pre-existing account or create one with Xilinx.  Sign into your Xilinx account or create a Xilinx account to download the web installer.  Once you sign in, go ahead and let your chosen installer download and then let it run.  Xilinx will ask you to sign in again (since you don’t necessarily have to install Vivado on the same computer you installed the web installer on) and choose how you want to download/install. I’ll choose the “Download and Install Now” to make I only download what I need to help conserve space on my laptop.  Agree to the license agreements and terms and conditions. I’m the type of person that actually looks through the license agreements so this took a bit of time for me.  Choose what version of the Xilinx’s Vivado Design Suite you wish to install. We’re going with the free version which is the Vivado HL WebPACK Edition, which is device limited to a smaller selection of FPGAs. Which FPGAs does this refer to? You can view a full list on page 9 of the [Vivado Design Suite User Guide](https://www.xilinx.com/support/documentation/sw_manuals/xilinx2016_4/ug973-vivado-release-notes-install-license.pdf" \t "_blank) by Xilinx, but in terms of Digilent boards, the 2016.4 WebPACK edition can target the [Zybo](http://store.digilentinc.com/zybo-zynq-7000-arm-fpga-soc-trainer-board/" \t "_blank), [ZedBoard](http://store.digilentinc.com/zedboard-zynq-7000-arm-fpga-soc-development-board/" \t "_blank), [PYNQ-Z1](http://store.digilentinc.com/pynq-z1-python-productivity-for-zynq/), both flavors of the [Cmod A7](http://store.digilentinc.com/cmod-a7-breadboardable-artix-7-fpga-module/" \t "_blank), [Arty](http://store.digilentinc.com/arty-board-artix-7-fpga-development-board-for-makers-and-hobbyists/), [Basys 3](http://store.digilentinc.com/basys-3-artix-7-fpga-trainer-board-recommended-for-introductory-users/" \t "_blank), [Nexys 4 DDR](http://store.digilentinc.com/nexys-4-ddr-artix-7-fpga-trainer-board-recommended-for-ece-curriculum/" \t "_blank), [Nexys Video](http://store.digilentinc.com/nexys-video-artix-7-fpga-trainer-board-for-multimedia-applications/" \t "_blank), and eventually the [Arty Z7](http://store.digilentinc.com/arty-z7-apsoc-zynq-7000-development-board-for-makers-and-hobbyists/) (when it is released). I happen to have the Arty, so the WebPACK edition will be perfect for me. It doesn’t say so on this page, but we can choose to add the Xilinx SDK with the installation. Note that you can compare how the WebPACK version compares with the other versions of Vivado on Xilinx’s website [here](https://www.xilinx.com/products/design-tools/vivado/vivado-webpack.html).  The next page lets you choose some specifics of what to install with the Vivado HL WebPACK edition. I currently plan on just using the [Arty](http://store.digilentinc.com/arty-board-artix-7-fpga-development-board-for-makers-and-hobbyists/) which uses an Artix 7 35t FPGA, so I’ll go ahead and un-check the boxes that don’t relate to the Artix-7 chip which include the [Zynq-7000](http://store.digilentinc.com/fpga-programmable-logic/by-technology/zynq/), [Kintex-7](http://store.digilentinc.com/fpga-programmable-logic/by-technology/kintex/), and Kintex Ultrascale, which saves me a little over 3 GB of disk space. I’ll go ahead and un-check the DocNav as well since I’m confident I’ll always have access to the internet to browse through the Xilinx documentation if I need to.  You get once last chance to panic and change your mind before hitting install here, but we were extra careful during the installation process, so we should be good to go. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date:3/6/2020** |  | **Name:** | **Shilpa .C** | |
| **Course:python** |  | **USN:** | **4al17ec086** | |
| **Topic: Application 7: Scrape Real Estate Property Data from the Web**  **Application 8: Build a Web-based Financial Graph** |  | **Semester & Section:** | **6th , B sec** | |
| **AFTERNOON SESSION DETAILS** | | | |
| **Image of session** | | | |
| **Report – Report can be typed or hand written for up to two pages.**  **Scrape Real Estate Property Data from the Web**  **Web Scraping in real estate to the rescue**  [**Web scraping**](https://blog.datahut.co/web-scraping-at-large-data-extraction-challenges-you-must-know/)is the process of sorting through overwhelming amounts of data, refine the user’s searches and provide a list of relevant information. In a realtor’s case, it is the go-to tool for organized property listings. Scraping the web provides parameters which the realtor can further study to determine sales and prospective buyers. Parameters extracted by web scraping are:   * Size * Property type * Location * Sale price * Size * Amenities * Monthly rental price * Parking spaces * Agent contact   This information is displayed in form of a spreadsheet, allowing the realtor to make comparisons of relevant parameters.   1. **Property value tracking**   Let’s assume you decide to sell your property. Scraping the web for the value of similar properties can aid you in setting a good value on your own. This allows users searching for such properties to get fair deals, and on the other, you getting a profitable one.  **2. Making the right investment**  Obtaining real estate data is hard, as result of which most [**investors make financial investments**](https://blog.datahut.co/data-scraping-investment-returns/)blindly. ith web scraping, an investor can make decisions based on qualitative and relevant empirical data, rather than outdated or incomplete information. Aggregating property data from real estate listing websites is essential for investment analysis.  **3. Rental Yield**  Rental yield is the most important factor to be considered before investing in property. By scraping data from real estate websites, you can determine which properties have the best rental yield for any suburb. Moreover, scraping answers which property types (house, apartment, 1 bedroom, 2 bedrooms) are more preferred in a particular area and yield the best return on investment.  **4. Track vacancy rates**  A vacant investment property is risky. To minimize this risk, it is imperative to analyze property data and suburbs which have higher rental listings.  The above parameters are the most relevant decoded by [**web scraping**](https://blog.datahut.co/choose-web-scraping-service-pricing-vs-cost-dilemma/) through numerous websites online. Having the above details at your fingertips improves a realtor’s efficacy at decision making, better communication and faster and profitable sales. The role of web scraping in retail is just getting started, its potential is however limitless.  **Build a Web-based Financial Graph**  Firstly, web scraping is performed to scrape the required data. We are using real-time stock market data for scraping and then store it into a CSV file format using Python libraries. Data is extracted from the web using Python's Beautiful Soup module. Beautiful Soup is an inbuilt package in Python that is used for parsing HTML and XML documents (including having distorted or abnormal mark- up, i.e. non-closed tags, so named after tag soup). It develops a parse tree for already parsed documents that can further be used to extract data from HTML file or document, which is useful in case of web scraping.  Secondly, the graph is a plot on our designed website from the CSV files containing the scraped data from the web. This task is done by using c3.js and Papa Parse libraries and their respective dependencies. C3 provides an easy way to construct D3-based charts by encapsulating the code that is required to generate the entire chart/graph. C3 library of JavaScript provides a wide range of APIs and Callbacks (Callback is a function in JavaScript that is executed after another program has finished its execution) to access the status of the chart at a particular time. By using this C3 library, we can update the chart/graph even after it is accomplished.  Another library that we have used in graph making is Papa Parse which is the fastest in-browser CSV (or delimited text) parser for JavaScript. Papa Parse is the world's first multi- threaded CSV parser used for the browser. It is reliable and easy to use. Papa-parse is an effective and convenient CSV parser that can handle files having size in gigabytes without crashing. It is capable of manipulating your CSV files in many ways. First off, the input. This component can read your data from anywhere, via a URL, from a raw string or even from your local storage. The output will be an array of rows, where each row is an array of table data, and it will be returned if the header flag is not set. Otherwise, an array of objects will be the product, where each object is a map comprising of the column name and its corresponding value or the row (e.g., {col1: value1, col2: value2}). The recent format is in the vicinity of a JSON file. | | | |