REPORT JUNE 10

Date:	**10 JUNE 2020**	**Name:**	**Rakshith B**
Course:	**Kicad on Udemy**	**USN:**	**4AL16EC409**
Topic:	**A hands –on tour of kiCAD with a simple project: schematic Design**	**Semester & Section:**	**6th SEM B**
Github Repository:	**Rakshith-B**		

FORENOON SESSION DETAILS
Image of session
Report – **what is PCB ?** **A printed circuit board (PCB) mechanically supports and electrically connects electrical or electronic components using conductive tracks, pads and other features etched from one or more sheet layers of copper laminated onto and/or between sheet layers of a non-conductive substrate.** ● **The process of designing a PCB using Kicad begins with Eeschema.** ● **The process begins with Eescema. In Eeschema we create the electrical schematic that describes the circuit that eventually will be printed onto the PCB board. We draw the schematic by picking components from the library and if a component that we need doesn't exist in the library, we can create it using the schematic library editor.** ● **At some point in our design of this schematic we can run the electrical rules check and that is a utility that ensures that we don't have any obvious mistakes or any electrical mistakes**

in our design. The electrical rules check will give us a defect report and we'll use that report correct any problems in Eeschema.

- **Once we have everything set out correctly and there are no more defects in our schematic, we are almost ready to move ahead and go into the Pcbnew application in order to start doing the layout for our PCB.**
- **There are two things we need to do before we go to Pcbnew. The first one is to associate the components in Eeschema with footprints. To do that we use another component of Kicad called Cvpcb.**

The basic workflow in KiCad is:

1. **Create a project.**
2. **Create a schematic with 'eeschema'.**
3. **Assign footprints to symbols and generate the netlist.**
4. **Create a board with 'pcbnew', importing the netlist from 'eeschema'.**
5. **Test the board using the 'Design Rule Check'.**
6. **Generate production files.**

Schema for the first time

- **Under Windows run kicad.exe. Under Linux type kicad in your Terminal. You are now in the main window of the KiCad project manager. From here you have access to eight stand-alone software tools: Eeschema, Schematic Library Editor, Pcbnew, PCB Footprint Editor, GerbView, Bitmap2Component, PCB Calculator and Pl Editor. Refer to the work-flow chart to give you an idea how the main tools are used.**
- **Create a new project: File → New → Project. Name the project file tutorial1. The project file will automatically take the extension ".pro". The exact appearance of the dialog depends on the used platform, but there should be a checkbox for creating a new directory. Let it stay checked unless you already have a dedicated directory. All your project files will be saved there.**
- **Let's begin by creating a schematic. Start the schematic editor Eeschema, . It is the first button from the left.**
- **Click on the Page Settings icon on the top toolbar. Set the appropriate paper size (A4,8.5x11 etc.) and enter the Title as Tutorial1. You will see that more information can be entered here if necessary. Click OK. Getting Started in KiCad 10 / 46 This information will populate the schematic sheet at the bottom right corner. Use the mouse wheel to zoom in. Save the whole schematic: File → Save**

- **We will now place our first component. Click on the Place symbol icon in the right toolbar. You may also press the Add Symbol hotkey [a]**
- **Click on the middle of your schematic sheet. A Choose Symbol window will appear on the screen. We're going to place a resistor. Search / filter on the R of Resistor. You may notice the Device heading above the Resistor. This Device heading is the name of the library where the component is located, which is quite a generic and useful library**
- **Double click on it. This will close the Choose Symbol window. Place the component in the schematic sheet by clicking where you want it to be**
- **Click on the magnifier icon to zoom in on the component. Alternatively, use the mouse wheel to zoom in and zoom out. Press the wheel (central) mouse button to pan horizontally and vertically.**
- **Try to hover the mouse over the component R and press [r]. The component should rotate. You do not need to actually click on the component to rotate it.**
- **Right click in the middle of the component and select Properties → Edit Value. You can achieve the same result by hovering over the component and pressing [v]. Alternatively, [e] will take you to the more general Properties window. Notice how the right-click menu below shows the hotkeys for all available actions.**
- **The Edit Value Field window will appear. Replace the current value R with 1 k. Click OK.**
- **The schematic is now finished. We can now create a Netlist file to which we will add the footprint of each component. Click on the Generate netlist icon on the top toolbar. Click on the Generate Netlist button and save under the default file name.**
- **There are many more ways to add footprints to symbols. • Right click on a symbol → Properties → Edit Footprint • Double click on a symbol, or right click on a symbol → Properties → Edit Properties → Footprint • Tools → Edit Symbol Fields • Check Show footprint previews in symbol chooser in Eeschema's preferences and select the footprint when you select a new symbol to place**
- **From the KiCad project manager, click on the Pcb layout editor icon . You can also use the corresponding toolbar button from Eeschema. The Pcbnew window will open. If you get a message saying that a *.kicad_pcb file does not exist and asks if you want to create it, just click Yes. 2. Begin by entering some schematic information. Click on the Page settings icon on the top toolbar. Set the appropriate paper size (A4,8.5x11 etc.) and title as Tutorial1. 3. It is a good idea to start by setting the clearance and the minimum track width to those required by your PCB manufacturer. In general you can set the clearance to 0.25 and the minimum track width to 0.25. Click on the Setup → Design Rules menu. If it does not show already, click on the Net Classes Editor tab. Change the Clearance field at the top of the**

window to 0.25 and the Track Width field to 0.25 as shown below. Measurements here are in mm.

- **Trace around the outline of the board by clicking each corner in rotation. Finish your rectangle by clicking the first corner second time. Right click inside the area you have just traced. Click on Zones→'Fill or Refill All Zones'. The board should fill in with green and look something like this:**
- **Run the design rules checker by clicking on the Perform design rules check icon on the top toolbar. Click on Start DRC. There should be no errors. Click on List Unconnected. There should be no unconnected items. Click OK to close the DRC Control dialogue. 26. Save your file by clicking on File → Save. To admire your board in 3D, click on View → 3D Viewer**

The process described here can easily be repeated as many times as you need. Beside the Forward Annotation method described above, there is another method known as Backward Annotation. This method allows you to make modifications to your already routed PCB from Pcbnew and updates those modifications in your schematic and netlist file. The Backward Annotation method, however, is not that useful and is therefore not described here

Date:	**09 JUNE 2020**	**Name:RAKSHITH B**
Course:	**PHP & MYSQL On Udemy**	**USN:4AL16EC409**
Topic:	**MySQL Joins,PHP Errors and security,Building a template page,**	**Semester & Section:6 B**

AFTERNOON SESSION DETAILS
Image of session

Report –

Creating Database:

Introduction to MySQL join clauses

A relational database consists of multiple related tables linking together using common columns which are known as foreign key columns. Because of this, data in each table is incomplete from the business perspective.

For example, in the sample database, we have the orders and orderdetails tables that are linked using the orderNumber column:

To get complete orders' information, you need to query data from both orders and orderdetails tables.

That's why joins come into the play.

A join is a method of linking data between one (self-join) or more tables based on values of the common column between the tables.

MySQL supports the following types of joins:

1. **Inner join**
2. **Left join**
3. **Right join**
4. **Cross join**

To join tables, you use the cross join, inner join, left join, or right join clause for the corresponding type of join. The join clause is used in the SELECT statement appeared after the FROM clause.

Note that MySQL hasn't supported the FULL OUTER JOIN yet.

Setting up sample tables

First, create two tables called members and committees:

```
CREATE TABLE members (
   member_id INT AUTO_INCREMENT,
   name VARCHAR(100),
   PRIMARY KEY (member_id)
);

CREATE TABLE committees (
   committee_id INT AUTO_INCREMENT,
```

```
    name VARCHAR(100),
    PRIMARY KEY (committee_id)
);
```

Second, insert some rows into the tables members and committees :

```
INSERT INTO members(name)
VALUES('John'),('Jane'),('Mary'),('David'),('Amelia');

INSERT INTO committees(name)
VALUES('John'),('Mary'),('Amelia'),('Joe');
```

Third, query data from the tables members and committees:

```
SELECT * FROM members;
SELECT * FROM committees;
```

MySQL INNER JOIN clause

The inner join clause joins two tables based on a condition which is known as a join predicate.

The inner join clause compares each row from the first table with every row from the second table. If values in both rows cause the join condition evaluates to true, the inner join clause creates a new row whose column contains all columns of the two rows from both tables and include this new row in the final result set. In other words, the inner join clause includes only rows whose values match.

The following shows the basic syntax of the inner join clause that joins two tables table_1 and table_2:

```
SELECT column_list
FROM table_1
INNER JOIN table_2 ON join_condition;
```

If the join condition uses the equal operator (=) and the column names in both tables used for matching are the same, you can use the USING clause instead:

```
SELECT column_list
FROM table_1
INNER JOIN table_2 USING (column_name);
```

The following statement finds members who are also the committee members:

```
SELECT
    m.member_id,
    m.name member,
    c.committee_id,
    c.name committee
FROM
```

```
   members m
INNER JOIN committees c
        ON c.name = m.name;
```

Because the name columns are the same in both tables, you can use the USING clause as shown in the following query:

```
SELECT
   m.member_id,
   m.name member,
   c.committee_id,
   c.name committee
FROM
   members m
INNER JOIN committees c USING(name);
```

MySQL LEFT JOIN clause

Similar to an inner join, a left join also requires a join-predicate. When joining two tables using a left join, the concepts of left and right tables are introduced.

The left join selects data starting from the left table. For each row in the left table, the left join compares with every row in the right table. If the values in the two rows cause the join condition evaluates to true, the left join creates a new row whose columns contain all columns of the rows in both tables and includes this row in the result set.

If the values in the two rows are not matched, the left join clause still creates a new row whose columns contain columns of the row in the left table and NULL for columns of the row in the right table.

In other words, the left join selects all data from the left table whether there are matching rows exist in the right table or not. In case there is no matching rows from the right table found, NULLs are used for columns of the row from the right table in the final result set.

Here is the basic syntax of a left join clause that joins two tables:

```
SELECT column_list
FROM table_1
LEFT JOIN table_2 ON join_condition;
```

The left join also supports the USING clause if the column used for matching in both tables are the same:

```
SELECT column_list
FROM table_1
LEFT JOIN table_2 USING (column_name);
```

The following example uses the left join to join the members with the committees table:

```
SELECT
   m.member_id,
   m.name member,
   c.committee_id,
   c.name committee
FROM
   members m
LEFT JOIN committees c USING(name);
```

Error Reporting

Example #1 Attacking Variables with a custom HTML page

```
<form method="post" action="attacktarget?username=badfoo&amp;password=badfoo">
<input type="hidden" name="username" value="badfoo" />
<input type="hidden" name="password" value="badfoo" />
</form>
```

Example #2 Exploiting common debugging variables

```
<form method="post" action="attacktarget?errors=Y&amp;showerrors=1&amp;debug=1">
<input type="hidden" name="errors" value="Y" />
<input type="hidden" name="showerrors" value="1" />
<input type="hidden" name="debug" value="1" />
</form>
```

Regardless of the method of error handling, the ability to probe a system for errors leads to providing an attacker with more information.

For example, the very style of a generic PHP error indicates a system is running PHP. If the attacker was looking at an .html page, and wanted to probe for the back-end (to look for known weaknesses in the system), by feeding it the wrong data they may be able to determine that a system was built with PHP.

A function error can indicate whether a system may be running a specific database engine, or give clues as to how a web page or programmed or designed. This allows for deeper investigation into open database ports, or to look for specific bugs or weaknesses in a web page. By feeding different pieces of bad data, for example, an attacker can determine the order of authentication in a script, (from the line number errors) as well as probe for exploits that may be exploited in different locations in the script.

A filesystem or general PHP error can indicate what permissions the web server has, as well as the structure and organization of files on the web server. Developer written error code can aggravate this problem, leading to easy exploitation of formerly "hidden" information.

PHP Templating

We'll look at why using such an architecture is useful and what tools we can use to accomplish this. Here's what we'll cover:

1. **Learn some basic MVC concepts,**
2. **Review some popular templating libraries,**
3. **Play around with a small custom-made view class.**
4. **Explore the basics of using the Twig library.**

To fully benefit from this article, you should already know how to write and run your own PHP scripts on a Web server (i.e. using Apache).

```
<html>
  <head>
    <title>Info</title>
  </head>
  <body>
    <pre>
      User Information:
      Name: {$name|capitalize}
      Addr: {$address|escape}
      Date: {$smarty.now|date_format:"%b %e, %Y"}
    </pre>
  </body>
</html>
```

```
<html>
  <head><title>My first Twig template!</title></head>
  <body>
    My name is {{ name }}.
    My friends are:
    <ul>
    {% for person in friends %}
      <li>{{ person.firstname}} {{ person.lastname }}</li>
    {% endfor %}
    </ul>
  </body>
</html>
```

```
<?php
class MyView {
  protected $template_dir = 'templates/';
  protected $vars = array();
  public function __construct($template_dir = null) {
    if ($template_dir !== null) {
      // Check here whether this directory really exists
      $this->template_dir = $template_dir;
    }
  }
```

```
   public function render($template_file) {
      if (file_exists($this->template_dir.$template_file)) {
         include $this->template_dir.$template_file;
      } else {
         throw new Exception('no template file ' . $template_file . ' present in directory ' .
$this->template_dir);
      }
   }
   public function __set($name, $value) {
      $this->vars[$name] = $value;
   }
   public function __get($name) {
      return $this->vars[$name];
   }
}
?>
```

INDEX.PHTML: HTML TEMPLATE

```
<html>
   <body>
      Names of my friends:
      <ul>
      <?php foreach ($this->friends as $friend): ?>
         <li><?=$friend?></li>
      <?php endforeach; ?>
      </ul>
   </body>
</html>
```