Database code:

db.define\_table(**'sheet'**,
 Field(**'created\_on'**, **'date'**, default = request.now, update = request.now, writable = False, requires = IS\_DATE(format=(**'%m-%d-%Y'**))),
 Field(**'created\_by'**, **'reference auth\_user'**, default = auth.user\_id, update = auth.user\_id, writable = False),
 Field(**'title'**, **'string'**, length=50, required = True, requires = IS\_NOT\_EMPTY(), default=**'Standard Finger Tip'**),
 Field(**'sheet\_version'**, **'integer'**, length=5, default = 1, writable = False),
 Field(**'handed'**, requires=IS\_IN\_SET([**'Right'**, **'Left'**, **'Two Handed'**]), default = **'Right'**),
 Field(**'grip\_type'**, requires=IS\_IN\_SET([**'Standard'**, **'Finger Tip'**]), default = **'Finger Tip'**),
 Field(**'bridge'**, **'string'**, length=15),
 Field(**'pap1'**, **'string'**, length=30),
 Field(**'pap2'**, **'string'**, length=30),
 Field(**'oval\_start'**, **'string'**, length=15),
 Field(**'oval\_end'**, **'string'**, length=15),
 Field(**'oval\_left'**, **'string'**, length=15),
 Field(**'oval\_right'**, **'string'**, length=15),
 Field(**'angle'**, **'string'**, length=15),
 Field(**'vertical'**, **'string'**, length=15),
 Field(**'horizontal'**, **'string'**, length=15),
 Field(**'slug'**, **'string'**, length=15),
 Field(**'offset\_p'**, **'string'**, length=15),
 Field(**'clt'**, **'string'**, length=15),
 Field(**'axis\_tilt\_choice'**, requires = IS\_IN\_SET([**''**, **'Low'**, **'Medium'**, **'High'**]), default=**''**),
 Field(**'axis\_tilt\_number'**, **'string'**, length=15),
 Field(**'revs\_choice'**, length=15, requires = IS\_IN\_SET([**''**, **'Low'**, **'Medium'**, **'High'**]), default=**''**),
 Field(**'revs\_number'**, **'string'**, length=15),
 Field(**'speed\_choice'**, requires = IS\_IN\_SET([**''**, **'Low'**, **'Medium'**, **'High'**]), default=**''**),
 Field(**'speed\_number'**, **'string'**, length=15),
 Field(**'left\_reverse\_pitch'**, **'string'**, length=15),
 Field(**'left\_lateral\_pitch'**, **'string'**, length=15),
 Field(**'left\_forward\_pitch'**, **'string'**, length=15),
 Field(**'left\_hole\_size'**, **'string'**, length=15),
 Field(**'left\_insert\_size'**, **'string'**, length=15),
 Field(**'left\_grip\_type'**, **'string'**, required = True, requires = IS\_IN\_SET([**''**, **'Blend'**, **'Lift'**, **'Oval'**, **'Semi'**]), default=**''**),
 Field(**'right\_reverse\_pitch'**, **'string'**, length=15),
 Field(**'right\_lateral\_pitch'**, **'string'**, length=15),
 Field(**'right\_forward\_pitch'**, **'string'**, length=15),
 Field(**'right\_hole\_size'**, **'string'**, length=15),
 Field(**'right\_insert\_size'**, **'string'**, length=15),
 Field(**'right\_grip\_type'**, **'string'**, required = True, requires = IS\_IN\_SET([**''**, **'Blend'**, **'Lift'**, **'Oval'**, **'Semi'**]), default=**''**),
 Field(**'thumb\_reverse\_pitch'**, **'string'**, length=15),
 Field(**'thumb\_right\_lateral\_pitch'**, **'string'**, length=15),
 Field(**'thumb\_left\_lateral\_pitch'**, **'string'**, length=15),
 Field(**'thumb\_forward\_pitch'**, **'string'**, length=15),
 Field(**'thumb\_hole\_size'**, **'string'**, length=15),
 Field(**'thumb\_IT'**, **'string'**, required = True, requires = IS\_IN\_SET([**'Yes'**, **'No'**]), default=**'No'**),
 Field(**'thumb\_mold'**, **'string'**, required = True, requires = IS\_IN\_SET([**'Yes'**, **'No'**]), default=**'No'**),
 Field(**'thumb\_insert\_size'**, **'string'**, length=15),
 Field(**'overall\_left'**, **'string'**, length=15),
 Field(**'overall\_right'**, **'string'**, length=15),
 Field(**'center\_left'**, **'string'**, length=15),
 Field(**'center\_right'**, **'string'**, length=15),
 Field(**'cut\_left'**, **'string'**, length=15),
 Field(**'cut\_right'**, **'string'**, length=15),
 Field(**'normal\_ball\_weight'**, **'string'**, length=15),
 Field(**'throw\_type'**, required = True, requires = IS\_IN\_SET([**''**, **'Full Roller'**, **'Back-Up'**]), default=**''**),
 Field(**'is\_active'**,**'boolean'**, writable=False, readable=False, default=True),
 Field(**'customer\_id'**, **'reference customer'**,
 requires = IS\_IN\_DB(db, db.customer.id, **'%(first\_name)s %(last\_name)s'**)),
 format = **'%(title)s %(id)s'**)

Control code:

@auth.requires\_login()
**def** new\_sheet():
 this\_customer = db.customer(request.args(0,cast=int))
 db.sheet.customer\_id.default = this\_customer.id
 db.sheet.created\_on.default = request.now
 db.sheet.created\_by.default = auth.user\_id
 form=SQLFORM(db.sheet)
 **if** form.process(session=None, formname=**'fingerprint'**).accepted:
 response.flash=**'Thanks for filling out the form'** db.sheet\_note.insert(sheet\_id = form.vars.id)
 redirect(URL(**'view\_sheet'** + **'/'** + str(form.vars.id)))
 **return** dict(customer=this\_customer)

View code:

{{extend 'layout.html'}}
<!DOCTYPE **html** PUBLIC **"-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd"**>
<**html**>
 <**head**>
 <**meta content="text/html; charset=utf-8" http-equiv="content-type"**>
 <**script type="text/javascript"**>
 function draw(){
 var canvas = document.getElementById('tutorial');
 if (canvas.getContext){
 var context = canvas.getContext('2d');
 var width = 700;
 var height = 700;
 var scale = width/450;

 var fradius = 50;
 var tradius = 60;
 var lfHor = (width / 8) \* 3.875;
 var lfVer = (height / 6) \* 1.25;
 var rfHor = (width / 8) \* 6.375;
 var rfVer = (height / 6) \* 1.25;
 var thmbHor = (width / 8) \* 5.125;
 var thmbVer = (height / 6) \* 4.75;
 var middle = lfHor+((rfHor-lfHor)/2)

 context.canvas.width = width;
 context.canvas.height = height;

 //circles
 draw\_circle(context, lfHor, lfVer, fradius);
 draw\_circle(context, rfHor, rfVer, fradius);
 draw\_circle(context, thmbHor, thmbVer, tradius);
 draw\_circle(context, lfHor-(260), lfVer-(60), 25);
 draw\_circle(context, lfHor-(180), lfVer-(60), 25);

 //lines that bisect circle
 bisect(context, 135, lfHor, lfVer, fradius);
 bisect(context, 135, rfHor, rfVer, fradius);
 bisect(context, 135, thmbHor, thmbVer, tradius);

 //lines between circles
 draw\_line2Circle(context, lfHor, lfVer, thmbHor, thmbVer, fradius, tradius);
 draw\_line2Circle(context, rfHor, rfVer, thmbHor, thmbVer, fradius, tradius);
 draw\_line2Circle(context, lfHor, lfVer, rfHor, rfVer, fradius, fradius);

 //lines
 draw\_line(context, lfHor + fradius, lfVer - fradius, lfHor + fradius, lfVer + fradius);
 draw\_line(context, rfHor - fradius, rfVer - fradius, rfHor - fradius, rfVer + fradius);
 draw\_line(context, middle-(160), thmbVer-(240), middle-(80), thmbVer-(240));
 draw\_line(context, middle+(80), thmbVer-(240), middle+(160), thmbVer-(240));
 draw\_line(context, middle-(150), thmbVer-(190), middle-(70), thmbVer-(190));
 draw\_line(context, middle+(70), thmbVer-(190), middle+(150), thmbVer-(190));
 draw\_line(context, middle-(140), thmbVer-(140), middle-(60), thmbVer-(140));
 draw\_line(context, middle+(60), thmbVer-(140), middle+(140), thmbVer-(140));

 //arrows
 draw\_arrow(context, lfHor, lfVer, 0, fradius, scale);
 draw\_arrow(context, lfHor, lfVer, 180, fradius, scale);
 draw\_arrow(context, lfHor, lfVer, 270, fradius, scale);
 draw\_arrow(context, rfHor, rfVer, 0, fradius, scale);
 draw\_arrow(context, rfHor, rfVer, 90, fradius, scale);
 draw\_arrow(context, rfHor, rfVer, 180, fradius, scale);
 draw\_arrow(context, thmbHor, thmbVer, 0, tradius, scale);
 draw\_arrow(context, thmbHor, thmbVer, 90, tradius, scale);
 draw\_arrow(context, thmbHor, thmbVer, 180, tradius, scale);
 draw\_arrow(context, thmbHor, thmbVer, 270, tradius, scale);

 //rectangles
 draw\_rect(context, (20), (475), (90), (30)); //positive axis point left
 draw\_rect(context, (110), (475), (90), (30)); //positive axis point right
 draw\_rect(context, (10), (515), (70), (30)); //axis tilt number
 draw\_rect(context, (210), (515), (70), (30)); //axis tilt choice
 draw\_rect(context, (10), (555), (70), (30)); //revs number
 draw\_rect(context, (210), (555), (70), (30)); //revs choice
 draw\_rect(context, (10), (595), (70), (30)); //speed number
 draw\_rect(context, (210), (595), (70), (30)); //speed choice
 draw\_rect(context, (40), (155), (70), (30)); //oval start
 draw\_rect(context, (110), (155), (70), (30)); //oval end
 draw\_rect(context, (115), (215), (80), (30)); //angle
 draw\_rect(context, (115), (255), (80), (30)); //vert
 draw\_rect(context, (115), (295), (80), (30)); //horz
 draw\_rect(context, (115), (335), (80), (30)); //slug
 draw\_rect(context, (115), (375), (80), (30)); //offset
 draw\_rect(context, (115), (415), (80), (30)); //clt
 draw\_rect(context, thmbHor+(145), thmbVer-(115), (70), (30)); //thumb IT
 draw\_rect(context, thmbHor+(145), thmbVer-(75), (70), (30)); //thumb Mold
 drawRotatedRect(context, thmbHor+(45), thmbVer+(65), (70), (30), 45); //overall right
 drawRotatedRect(context, thmbHor-(115), thmbVer+(65), (70), (30), 135); //overall left

 //static text
 draw\_text(context, "POSITIVE AXIS POINT", (110), (470), "black", "center", "n", scale, 0);
 draw\_text(context, "AXIS TILT", (145), (540), "black", "center", "n", scale, 0);
 draw\_text(context, "REVOLUTIONS", (145), (580), "black", "center", "n", scale, 0);
 draw\_text(context, "SPEED", (145), (620), "black", "center", "n", scale, 0);
 draw\_text(context, "OVAL", (110), (150), "black", "center", "n", scale, 0);
 draw\_text(context, "OVAL", (145), (210), "black", "center", "n", scale, 0);
 draw\_text(context, "ANGLE:", (105), (240), "black", "right", "n", scale, 0);
 draw\_text(context, "VERT:", (105), (280), "black", "right", "n", scale, 0);
 draw\_text(context, "HORZ:", (105), (320), "black", "right", "n", scale, 0);
 draw\_text(context, "SLUG:", (105), (360), "black", "right", "n", scale, 0);
 draw\_text(context, "OFFSET:", (105), (400), "black", "right", "n", scale, 0);
 draw\_text(context, "CLT:", (105), (440), "black", "right", "n", scale, 0);
 draw\_text(context, "GRIP TYPE", lfHor-(220), lfVer-(90), "black", "center", "n", scale, 0);
 draw\_text(context, "LEFT", lfHor-(260), lfVer-(10), "black", "center", "n", scale, 0);
 draw\_text(context, "RIGHT", lfHor-(180), lfVer-(10), "black", "center", "n", scale, 0);
 draw\_text(context, "BRIDGE", middle, lfVer+30, "black", "center", "n", scale, 0);
 draw\_text(context, "R", lfHor, lfVer-(80), "black", "center", "n", scale, 0);
 draw\_text(context, "R", rfHor, rfVer-(80), "black", "center", "n", scale, 0);
 draw\_text(context, "F", lfHor, lfVer+(100), "black", "center", "n", scale, 0);
 draw\_text(context, "F", rfHor, rfVer+(100), "black", "center", "n", scale, 0);
 draw\_text(context, "L", lfHor-(90), lfVer+(10), "black", "center", "n", scale, 0);
 draw\_text(context, "L", rfHor+(90), lfVer+(10), "black", "center", "n", scale, 0);
 draw\_text(context, "F", thmbHor, thmbVer-(90), "black", "center", "n", scale, 0);
 draw\_text(context, "R", thmbHor, thmbVer+(110), "black", "center", "n", scale, 0);
 draw\_text(context, "L", thmbHor-(100), thmbVer+(10), "black", "center", "n", scale, 0);
 draw\_text(context, "R", thmbHor+(100), thmbVer+(10), "black", "center", "n", scale, 0);
 draw\_text(context, "OVERALL", thmbHor, thmbVer-(240), "black", "center", "n", scale, 0);
 draw\_text(context, "CENTER", thmbHor, thmbVer-(190), "black", "center", "n", scale, 0);
 draw\_text(context, "CUT", thmbHor, thmbVer-(140), "black", "center", "n", scale, 0);
 draw\_text(context, "THB IT:", thmbHor+(110), thmbVer-(90), "black", "center", "n", scale, 0);
 draw\_text(context, "THB MOLD:", thmbHor+(95), thmbVer-(50), "black", "center", "n", scale, 0);

 } //end if
 } // end draw

 function draw\_circle(context, center\_x, center\_y, radius) {
 var lineWidth = 3;
 context.beginPath();
 context.arc(center\_x, center\_y, radius, 0, 2 \* Math.PI, false);
 context.fillStyle = '#dcdcdc';
 context.fill();
 context.lineWidth = lineWidth;
 context.strokeStyle = '#003300';
 context.stroke();
 }

 function vec(x, y) {
 this.length = function () {
 return Math.sqrt((this.x \* this.x) + (this.y \* this.y));
 };
 this.normalize = function () {
 var scale = this.length();
 this.x /= scale;
 this.y /= scale;
 };
 this.x = x;
 this.y = y;
 }

 function draw\_line2Circle(context, center1\_x, center1\_y, center2\_x, center2\_y, fradius, tradius) {
 var lineWidth = 3;
 var betweenVec = new vec(center2\_x - center1\_x, center2\_y - center1\_y);
 betweenVec.normalize();

 var p1x = center1\_x + (fradius \* betweenVec.x);
 var p1y = center1\_y + (fradius \* betweenVec.y);

 var p2x = center2\_x - (tradius \* betweenVec.x);
 var p2y = center2\_y - (tradius \* betweenVec.y);

 context.lineWidth = lineWidth;
 context.beginPath();
 context.moveTo(p1x, p1y);
 context.lineTo(p2x, p2y);
 context.stroke();
 }

 function draw\_line(context, p1x, p1y, p2x, p2y) {
 var lineWidth = 3;
 context.lineWidth = lineWidth;
 context.beginPath();
 context.moveTo(p1x, p1y);
 context.lineTo(p2x, p2y);
 context.stroke();
 }

 function draw\_rect(context, p1x, p1y, w, h) {
 var lineWidth = 3;
 context.beginPath();
 context.rect(p1x, p1y, w, h);
 context.fillStyle = '#dcdcdc';
 context.fill();
 context.lineWidth = lineWidth;
 context.strokeStyle = 'black';
 context.stroke();
 }

 function drawRotatedRect(context, p1x, p1y, w, h, degrees){
 var lineWidth = 3;
 // first save the untranslated/unrotated context
 context.save();

 context.beginPath();
 // move the rotation point to the center of the rect
 context.translate( p1x+w/2, p1y+h/2 );
 // rotate the rect
 context.rotate(degrees\*Math.PI/180);

 // draw the rect on the transformed context
 // Note: after transforming [0,0] is visually [x,y]
 // so the rect needs to be offset accordingly when drawn
 context.rect( -w/2, -h/2, w,h);
 context.lineWidth = lineWidth;
 context.fillStyle="#dcdcdc";
 context.fill();
 context.strokeStyle = 'black';
 context.stroke();

 // restore the context to its untranslated/unrotated state
 context.restore();
 }

 function draw\_arrow(context, center1\_x, center1\_y, side, radius, scale) {
 var lineWidth = 3;
 var offset = .5;
 var length = 25;
 context.lineWidth = lineWidth;
 context.beginPath();

 if (side === 0) {
 var posX = (center1\_x + (radius \* offset));
 var posY = (center1\_y - (radius + 5));
 context.moveTo(posX, posY);
 context.lineTo(posX, posY - length);
 context.lineTo(center1\_x, posY - (length \* 2));
 context.lineTo((center1\_x - (radius \* offset)), posY - length);
 context.lineTo((center1\_x - (radius \* offset)), posY);
 context.closePath();
 } else if (side === 90) {
 posX = (center1\_x + (radius + 5));
 posY = (center1\_y + (radius \* offset));
 context.moveTo(posX, posY);
 context.lineTo(posX + length, posY);
 context.lineTo(posX + (length \* 2), center1\_y);
 context.lineTo(posX + length, (center1\_y - (radius \* offset)));
 context.lineTo(posX, (center1\_y - (radius \* offset)));
 context.closePath();
 } else if (side === 180) {
 posX = (center1\_x - (radius \* offset));
 posY = (center1\_y + (radius + 5));
 context.moveTo(posX, posY);
 context.lineTo(posX, posY + length);
 context.lineTo(center1\_x, posY + (length \* 2));
 context.lineTo((center1\_x + (radius \* offset)), posY + length);
 context.lineTo((center1\_x + (radius \* offset)), posY);
 context.closePath();
 } else if (side === 270) {
 posX = (center1\_x - (radius + 5));
 posY = (center1\_y - (radius \* offset));
 context.moveTo(posX, posY);
 context.lineTo(posX - length, posY);
 context.lineTo(posX - (length \* 2), center1\_y);
 context.lineTo(posX - length, (center1\_y + (radius \* offset)));
 context.lineTo(posX, (center1\_y + (radius \* offset)));
 context.closePath();
 }

 context.fillStyle = '#dcdcdc';
 context.fill();
 context.stroke();
 }

 function bisect(context, degrees, cx, cy, radius) {
 var lineWidth = 3;
 // calculate the point on the edge of the circle
 var x1 = cx + radius \* Math.cos(degrees / 180 \* Math.PI);
 var y1 = cy + radius \* Math.sin(degrees / 180 \* Math.PI);
 // get the point on the opposite side of the circle
 // e.g. if 90, get 270, and vice versa
 // (super verbose but easily readable)
 if (degrees > 180) {
 var degrees2 = degrees - 180;
 } else {
 var degrees2 = degrees + 180;
 }
 // and calculate the point on the opposite side
 var x2 = cx + radius \* Math.cos(degrees2 / 180 \* Math.PI);
 var y2 = cy + radius \* Math.sin(degrees2 / 180 \* Math.PI);
 // now actually draw the line
 context.beginPath();
 context.moveTo(x1, y1);
 context.lineTo(x2, y2);
 context.stroke();
 }

 function draw\_text(context, text, x, y, color, align, bold, scale, degrees) {
 var new\_size = 16;
 if (text=="None") {
 text='';
 }
 // save orientation again
 context.save();
 // hold top-right hand corner when rotating
 context.translate( x, y );
 // rotate 270 degrees
 context.rotate(degrees\*Math.PI/180);
 context.fillStyle = color;
 if (bold == "b") {
 context.font = 'bold ' + new\_size + 'px' + ' ' + 'Sans-Serif';
 } else {
 context.font = new\_size + 'px' + ' ' + 'Sans-Serif';
 }
 context.textAlign = align;
 context.textBaseline = "bottom";
 context.fillText(text, 0, 0);
 context.restore();
 }
 </**script**>
 </**head**>
 <**body onload="draw();"**>
 <**form enctype="multipart/form-data" action="#" method="post"**>
 <**div class="row" padding="0"**>
 <**div class="col-md-9" style="**position:relative;width:720px;height:720px**"**>
 <**canvas id="tutorial" style="**border:3px solid #000000;**"**></**canvas**>
 <**input type="text" style="**position:absolute;left:315px;top:285px;width:60px;**" name="overall\_left"**>
 <**input type="text" style="**position:absolute;left:555px;top:285px;width:60px;**" name="overall\_right"**>
 <**input type="text" style="**position:absolute;left:325px;top:335px;width:60px;**" name="center\_left"**>
 <**input type="text" style="**position:absolute;left:545px;top:335px;width:60px;**" name="center\_right"**>
 <**input type="text" style="**position:absolute;left:335px;top:385px;width:60px;**" name="cut\_left"**>
 <**input type="text" style="**position:absolute;left:535px;top:385px;width:60px;**" name="cut\_right"**>
 <**input type="text" style="**position:absolute;left:435px;top:115px;width:60px;**" name="bridge"**>
 <**input type="text" style="**position:absolute;left:337px;top:65px;width:40px;**" name="left\_reverse\_pitch"**>
 <**input type="text" style="**position:absolute;left:261px;top:135px;width:40px;**" name="left\_lateral\_pitch"**>
 <**input type="text" style="**position:absolute;left:337px;top:207px;width:40px;**" name="left\_forward\_pitch"**>
 <**input type="text" style="**position:absolute;left:325px;top:115px;width:40px;**" name="left\_hole\_size"**>
 <**input type="text" style="**position:absolute;left:355px;top:155px;width:40px;**" name="left\_insert\_size"**>
 <**input type="text" style="**position:absolute;left:555px;top:65px;width:40px;**" name="right\_reverse\_pitch"**>
 <**input type="text" style="**position:absolute;left:632px;top:135px;width:40px;**" name="right\_lateral\_pitch"**>
 <**input type="text" style="**position:absolute;left:555px;top:207px;width:40px;**" name="right\_forward\_pitch"**>
 <**input type="text" style="**position:absolute;left:540px;top:115px;width:40px;**" name="right\_hole\_size"**>
 <**input type="text" style="**position:absolute;left:575px;top:155px;width:40px;**" name="right\_insert\_size"**>
 <**input type="text" style="**position:absolute;left:445px;top:625px;width:40px;**" name="thumb\_reverse\_pitch"**>
 <**input type="text" style="**position:absolute;left:532px;top:545px;width:40px;**" name="thumb\_right\_lateral\_pitch"**>
 <**input type="text" style="**position:absolute;left:360px;top:545px;width:40px;**" name="thumb\_left\_lateral\_pitch"**>
 <**input type="text" style="**position:absolute;left:445px;top:463px;width:40px;**" name="thumb\_forward\_pitch"**>
 <**input type="text" style="**position:absolute;left:425px;top:523px;width:40px;**" name="thumb\_hole\_size"**>
 <**input type="text" style="**position:absolute;left:470px;top:563px;width:40px;**" name="thumb\_insert\_size"**>
 <**select style="**position:absolute;left:65px;top:77px**" name="left\_grip\_type"**>
 <**option value=""**></**option**>
 <**option value="Blend"**>Blend</**option**>
 <**option value="Lift"**>Lift</**option**>
 <**option value="Oval"**>Oval</**option**>
 <**option value="Semi"**>Semi</**option**>
 </**select**>
 <**select style="**position:absolute;left:145px;top:77px**" name="right\_grip\_type"**>
 <**option value=""**></**option**>
 <**option value="Blend"**>Blend</**option**>
 <**option value="Lift"**>Lift</**option**>
 <**option value="Oval"**>Oval</**option**>
 <**option value="Semi"**>Semi</**option**>
 </**select**>
 <**input type="text" style="**position:absolute;left:40px;top:480px;width:85px;**" name="pap1"**>
 <**input type="text" style="**position:absolute;left:130px;top:480px;width:85px;**" name="pap2"**>
 <**input type="text" style="**position:absolute;left:62px;top:160px;width:60px;**" name="oval\_start"**>
 <**input type="text" style="**position:absolute;left:132px;top:160px;width:60px;**" name="oval\_end"**>
 <**input type="text" style="**position:absolute;left:142px;top:220px;width:60px;**" name="angle"**>
 <**input type="text" style="**position:absolute;left:142px;top:260px;width:60px;**" name="vertical"**>
 <**input type="text" style="**position:absolute;left:142px;top:300px;width:60px;**" name="horizontal"**>
 <**input type="text" style="**position:absolute;left:142px;top:340px;width:60px;**" name="slug"**>
 <**input type="text" style="**position:absolute;left:142px;top:380px;width:60px;**" name="offset\_p"**>
 <**input type="text" style="**position:absolute;left:142px;top:420px;width:60px;**" name="clt"**>
 <**select style="**position:absolute;left:30px;top:522px**" name="axis\_tilt\_choice"**>
 <**option value=""**></**option**>
 <**option value="Low"**>Low</**option**>
 <**option value="Medium"**>Medium</**option**>
 <**option value="High"**>High</**option**>
 </**select**>
 <**input type="text" style="**position:absolute;left:232px;top:520px;width:60px;**" name="axis\_tilt\_number"**>
 <**select style="**position:absolute;left:30px;top:562px**" name="revs\_choice"**>
 <**option value=""**></**option**>
 <**option value="Low"**>Low</**option**>
 <**option value="Medium"**>Medium</**option**>
 <**option value="High"**>High</**option**>
 </**select**>
 <**input type="text" style="**position:absolute;left:232px;top:560px;width:60px;**" name="revs\_number"**>
 <**select style="**position:absolute;left:30px;top:602px**" name="speed\_choice"**>
 <**option value=""**></**option**>
 <**option value="Low"**>Low</**option**>
 <**option value="Medium"**>Medium</**option**>
 <**option value="High"**>High</**option**>
 </**select**>
 <**input type="text" style="**position:absolute;left:232px;top:600px;width:60px;**" name="speed\_number"**>
 <**select style="**position:absolute;left:620px;top:447px**" name="thumb\_IT"**>
 <**option value="No"**>No</**option**>
 <**option value="Yes"**>Yes</**option**>
 </**select**>
 <**select style="**position:absolute;left:620px;top:487px**" name="thumb\_mold"**>
 <**option value="No"**>No</**option**>
 <**option value="Yes"**>Yes</**option**>
 </**select**>
 <**input type="text" style="**position:absolute;left:365px;top:625px;width:40px;**" name="oval\_left"**>
 <**input type="text" style="**position:absolute;left:530px;top:625px;width:40px;**" name="oval\_right"**>
 </**div**>
 <**div class="col-md-3"**>
 <**div class="row"**>
 <**div class="col-md-12"**>
 <**h3**>
 Customer Information
 {{=customer.first\_name}} {{=customer.last\_name}}
 </**h3**>
 Title: <**input type="text" style="**position:relative;width:200px;**" name="title"**><**br**/>
 Grip Type: <**select name="grip\_type"**>
 <**option value="Standard"**>Standard</**option**>
 <**option value="Finger Tip"**>Finger Tip</**option**>
 </**select**><**br**/>
 Handed: <**select name="handed"**>
 <**option value="Right"**>Right</**option**>
 <**option value="Left"**>Left</**option**>
 <**option value="Two Handed"**>Two Handed</**option**>
 </**select**><**br**/>
 Throw Type: <**select name="handed"**>
 <**option value=""**></**option**>
 <**option value="Full Roller"**>Full Roller</**option**>
 <**option value="Back-Up"**>Back-Up</**option**>
 </**select**><**br**/>
 Normal Ball Weight: <**input type="text" style="**position:relative;width:60px;**" name="normal\_ball\_weight"**><**br**/>
 <**input type="submit" style="**position:relative**" class="btn btn-primary" value="Submit"**/>
 <**input type="hidden" name="sheet\_version" value="1"**/>
 <**input type="hidden" name="created\_on" value="{{=request.now.strftime("%m-%d-%Y**")}}"/>
 <**input type="hidden" name="created\_by" value="{{=auth.user\_id}}"**/>
 <**input type="hidden" name="\_formname" value="fingerprint"**/>
 </**div**>
 </**div**>
 </**div**>
 </**div**>
 </**form**>
 <**h2**>Submitted variables</**h2**>
 {{=BEAUTIFY(request.vars)}}

 </**body**>
</**html**>