Topic	BARCODE MARKER BASED AR	
Class Description	**Students will learn about barcode markers. Students will also learn to create Augmented reality based on barcode markers.**	
Class	**C174**	
Class time	**45 mins**	
Goal	• Learn to use barcode markers for web based Augmented reality in A-Frame. • Learn to show atomic molecules in Augmented reality using barcode markers.	
Resources Required	• Teacher Resources: ○ Visual Studio Code Editor ○ laptop with internet connectivity ○ smartphone ○ earphones with mic ○ notebook and pen • Student Resources: ○ Visual Studio Code Editor ○ laptop with internet connectivity ○ smartphone ○ earphones with mic ○ notebook and pen	
Class structure	**Warm-Up** **Teacher-led Activity** **Student-led Activity** **Wrap-Up**	**5 mins** **15 mins** **20 mins** **5 mins**
• **WARM UP SESSION - 5 mins**		
CONTEXT • **Barcode maker based AR in A-Frame.**		

Teacher Starts Slideshow  **Slide 1 to 3** Refer to speaker notes and follow the instructions on each slide.	
Hey <student's name>. How are you? It's great to see you! Are you excited to learn something new today? **Following are the WARM-UP session deliverables:** • Greet the student. • Revision of previous class activities. • Quizzes.	**ESR**: Hi, thanks! Yes I am excited about it! Click on the slide show tab and present the slides
WARM-UP QUIZ Click on In-Class Quiz	
Continue WARM-UP Session  **Slide 4 to 16**	
Following are the session deliverables: • Appreciate the student. • Narrate the story by using hand gestures and voice modulation methods to bring in more interest in students.	

Class Steps	Teacher Action	Student Action
Step 1: Warm-Up (5 mins)	In the past few classes we have made a useful Augmented reality application for menu cards available at restaurants.	

	It would be really amazing if we could see such augmented reality cards menu cards at restaurants in the future because I would love to see the food out on the table before it's cooked to be served, it can help us decide on ordering the food in a better way. And of course I won't hesitate to try a new food if I can see it before, right?	**ESR**: Yes!
	Augmented reality is evolving day by day in various fields and one of the most popular one is education.	**ESR**: Yes.
	Now we don't have to limit ourselves to just reading books and trying to understand what it could be. Sometimes at school, for science subjects, if you could actually see what the teacher	**ESR**: Yes.

	is trying to explain, that would be super cool, don't you think? Today we are going to start making an augmented reality web based app where we can actually see the formation of chemical compounds. Are you excited?	
	Let's get started then.	
Teacher Ends Slideshow 		
TEACHER-LED ACTIVITY - 15 mins		
Teacher Initiates Screen Share		
CHALLENGE • **Use Barcode Markers for Augmented reality.** • **Make Augmented reality chemical atoms in A-Frame.**		

Step 2: Teacher-led Activity (15 mins)	Before we can make this application, we should be aware of a few things. This application is specific to the chemical compound formation; we should know how the compounds are formed. Can you tell me what chemical compounds are? Great! Everything around us is made up of **atoms**; these are the smallest particles that exist in nature. Atoms combine together to form molecules and compounds. Diagrammatically we can represent these with the small circles in 2D and with spheres in 3D.	**ESR**: Chemical compounds are formed by the combination of two or more atoms of different elements.

	Also the atom is made up of three particles called **neutrons**, **protons** and **electrons**. Neutrons and protons combined together is called the **nucleus** of the atom, which is at the centre of the atom. And **electrons revolve around the nucleus** in a certain configuration, called **atomic orbitals**. This representation of atoms, with the nucleus, electrons and atomic orbitals, of elements is known as the **atomic structure** of the elements. Some electrons are free to move out of these orbitals when they try to form compounds. And you might be wondering that if an	

	atom also has three particles then why do we still call it the smallest particle that can exist. Well the reason is we cannot break it down further into these particles. In today's class, we are going to start making these atomic structures of some elements in Augmented reality. Later on, in the upcoming class, we will see the formation of the compounds using these atomic structures in Augmented reality. ***Note**: Encourage the student to read more about the atomic structure and compound formation on their own before the next class for better understanding.*	

	We are going to make a marker based application again but we are going to use the different kinds of markers that are called **barcode markers**. These markers are special kinds of markers which have unique ID values assigned to them starting with 0. There are different types of barcode markers based on the way the **detection information** has been stored in the form of **matrix**. The most popular are 3X3 and 4X4 matrix code type barcode markers. *<The teacher opens the link to show images of some of the 3x3 Barcode Markers.>*	

	We don't have to go into the depth of creating these markers and can directly use them with the specific ID in our application. Let's begin! *<The teacher clones the code from the Teacher Activity 1.>* ***[Teacher Activity 1]*** We have the basic AR scene. Now we will have to specify the detection mode and matrix code type for these barcode markers to be able to render the objects over them. *<The teacher specifies the detection mode in <a-scene> in index.html.>*	

```
<head>
    <meta charset="utf-8" />
    <title>AR Chemistry</title>
    <meta name="viewport" content="width=device-width, user-scalable=no, minimum-scale=1.0, maximum-scale=1.0" />
    <script src="https://aframe.io/releases/1.0.1/aframe.min.js"></script>
    <script src="https://rawgit.com/jeromeetienne/AR.js/master/aframe/build/aframe-ar.min.js"></script>
    <script src="./js/Compounds.js"></script>
</head>

<body style="margin : 0px; overflow: hidden;">

    <!-- Add detectionMode and matrixCodeType to tell AR.js to recognize barcode markers -->

    <a-scene id="main-scene" vr-mode-ui="enabled: false"
        arjs="sourceType: webcam; debugUIEnabled: false; detectionMode: mono_and_matrix; matrixCodeType: 3x3;" embedded>
        <a-entity id="camera" camera></a-entity>

    </a-scene>
</body>
```

	Now we are going to register an A-Frame component, **atoms**, to create Augmented reality atoms of a few elements present in the periodic table. Before that, can you tell me what a periodic table is? The periodic table shows the elements with their chemical symbol and properties like their mass and number of electrons in the tabular structure. *<The teacher explains a little bit*	**ESR**: Varied.

	about the periodic table.> To begin with, we are going to pick up **sodium** (Na) and **Chlorine** (Cl), which forms **sodium chloride** (NaCl) also, known as **salt** that you use daily in your food. For each element, we are going to use different barcode marker numbers. We should keep the information of elements like name, barcode value, number of electrons separate. In the earlier app we used a database to keep pattern markers, models etc. Here we will create a separate JSON file in the directory itself to keep this information. We are also going to have a JSON file to	

	keep different colors for each element. *<The teacher creates a compoundList.json & elementColors.json files and adds data about the Na & Cl element.>*	
	  	
	Now we are going to write functions in the **atoms** component and attach in the scene:	

	• **getCompounds()** to get the element data: name, number of electrons and barcode value. • **getElementColors()** to get the color of the element, and • **createAtoms()** to create the 3D atomic structure (nucleus and free electrons) of the elements.	

```
AFRAME.registerComponent("atoms", {
    init: async function() {


    },
    getCompounds: function() {
      //Get compound details from the JSON file


    },
    getElementColors: function() {
      //Get color of the element from the JSON file



    },
    createAtoms: async function(element) {


      //Add BARCODE marker


      //Add atom card


      //Add nucleus of the


      //Create orbit for electron rvolution


      //Create revolving electrons

    }
  });
```

```
<!-- Add detectionMode and matrixCodeType to tell AR.js
<a-scene id="main-scene" vr-mode-ui="enabled: false"
  arjs="sourceType: webcam; debugUIEnabled: false;
  detectionMode: mono_and_matrix; matrixCodeType: 3x3;"
  embedded
  atoms>
  <a-entity id="camera" camera></a-entity>
</a-scene>
```

	Let's begin by writing functions to fetch details of the elements and colors. To do that we will use the **fetch()** function to get the details from the JSON file as we have done in earlier classes.	
	 ``` getCompounds: function() { return fetch("js/compoundList.json") .then(res => res.json()) .then(data => data); }, getElementColors: function() { return fetch("js/elementColors.json") .then(res => res.json()) .then(data => data); }, ```	
	Call **getCompounds()** to fetch the data of elements. This data can be passed while we call the **createAtoms()** function.	

```
init: async function () {

  //Get the compund details of the element
  var compounds = await this.getCompounds();

  var barcodes = Object.keys(compounds);

  barcodes.map(barcode => {
    var element = compounds[barcode];

    //Call the function
    this.createAtoms(element);
  });

},
```

Now first, let's take the values of the element name, barcode value and a number of electrons into the separate variables **elementName, barcodeValue** and **numOfElectron**.

We will also get the colors of the element by calling **getElementColors().**

Then we can create the **marker**, **atoms**, **nucleus**, **orbit** and **electrons** entity and add these in the scene.

	To add the **marker** entity: • Create an 'a-marker' element using **document.createElement()**. • Set the **id**, **type**, **element_name** and **value** (this attribute is for the barcode unique id value assigned to them) attribute using **.setAttribute()**. • Append the marker entity to the scene using **.appendChild()**.	

```
//Element data
var elementName = element.element_name;
var barcodeValue = element.barcode_value;
var numOfElectron = element.number_of_electron;

//Get the color of the element
var colors = await this.getElementColors();

//Scene
var scene = document.querySelector("a-scene");

//Add marker entity for BARCODE marker
var marker = document.createElement("a-marker");

marker.setAttribute("id", `marker-${barcodeValue}`);
marker.setAttribute("type", "barcode");
marker.setAttribute("element_name", elementName);
marker.setAttribute("value", barcodeValue);

scene.appendChild(marker);
```

To add the **atom** entity:

- Create an 'a-entity' element using **document.createElement()**.
- Set the **id** attribute using **.setAttribute()** .
- Append the entity to the marker using

	.appendChild (). To add the **atom card** entity: • Create an 'a-entity' element using **document.createElement()**. • Set the **id**, **geometry**, **position**, **rotation**, **material** attribute using **.setAttribute()** . • Append the card entity to the atom using **.appendChild ()**.	

```
var atom = document.createElement("a-entity");
atom.setAttribute("id", `${elementName}-${barcodeValue}`);
marker.appendChild(atom);

//Add atom card
var card = document.createElement("a-entity");
card.setAttribute("id", `card-${elementName}`);
card.setAttribute("geometry", {
  primitive: "plane",
  width: 1,
  height: 1
});

card.setAttribute("material", {
  src: `./assets/atom_cards/card_${elementName}.png`
});

card.setAttribute("position", { x: 0, y: 0, z: 0 });
card.setAttribute("rotation", { x: -90, y: 0, z: 0 });

atom.appendChild(card);
```

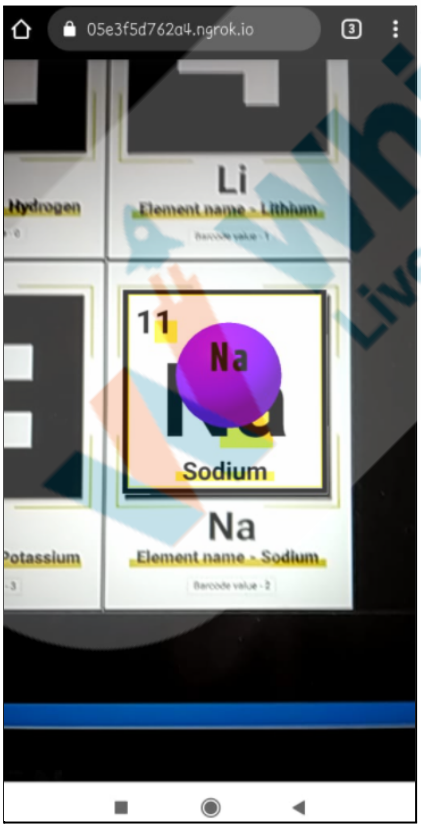
	To add the **nucleus** entity: • Create an 'a-entity' element using **document.createElement()**. • Set the **id**, **geometry**, **position**, **rotation**, **material color** attribute using **.setAttribute()**. • Append the entity to the atom using **.appendChild()**. To add the **nucleus name** (text to show the element symbol on the sphere) entity: • Create an 'a-entity' element using **document.createElement()**. • Set the **id**, **position**, **rotation**, **text** attribute using	

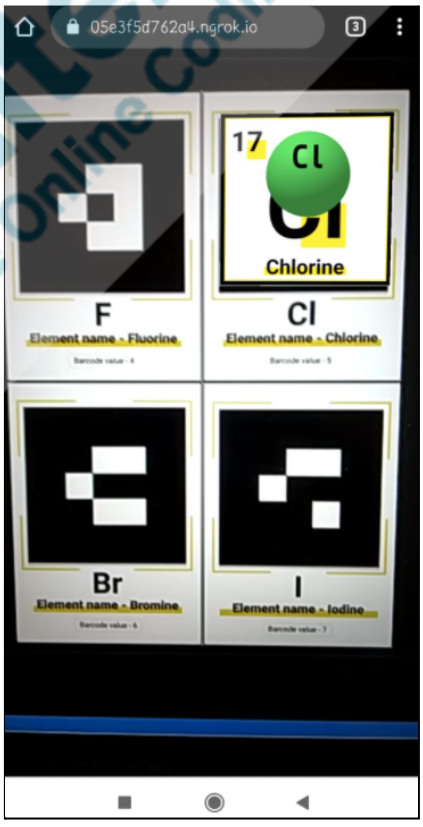
	.setAttribute() . • Append the entity to the nucleus using **.appendChild ()**.	

```
//Add nucleus
var nucleusRadius = 0.2;
var nucleus = document.createElement("a-entity");
nucleus.setAttribute("id", `nucleus-${elementName}`);
nucleus.setAttribute("geometry", {
  primitive: "sphere",
  radius: nucleusRadius
});

nucleus.setAttribute("material", "color", colors[elementName]);
nucleus.setAttribute("position", { x: 0, y: 1, z: 0 });

nucleus.setAttribute("rotation", { x: 0, y: 0, z: 0 });

var nucleusName = document.createElement("a-entity");
nucleusName.setAttribute("id", `nucleus-name-${elementName}`);
nucleusName.setAttribute("position", { x: 0, y: 0.21, z: -0.06 });
nucleusName.setAttribute("rotation", { x: -90, y: 0, z: 0 });
nucleusName.setAttribute("text", {
  font: "monoid",
  width: 3,
  color: "black",
  align: "center",
  value: elementName
});

nucleus.appendChild(nucleusName);

atom.appendChild(nucleus);
```

	Now we can test the output using ngrok.	

	***Note 1**: The barcode marker of id 2 is for Na and id 5 is Cl.* ***Note 2:** Find the barcode marker images available in the assets folder of the repository.* ***Note 3:** Scan (Point the camera towards) the marker image to see the output.*	





	We learned to use barcode markers to create atoms in the AR scene. Now you will add the electrons revolving around them. Are you excited?	**ESR**: Yes!
Teacher Stops Screen Share		
	Now it's your turn. Please share your screen with me.	
Teacher Starts Slideshow  **Slide 18 to 20** Refer to speaker notes and follow the instructions on each slide.		
We have one more class challenge for you. Can you solve it? Let's try. I will guide you through it.		
Teacher Ends Slideshow 		
STUDENT-LED ACTIVITY - 20 mins		
• **Ask the student to press the ESC key to come back to the panel.** • **Guide the student to start screen share.** • **Teacher gets into fullscreen.**		

ACTIVITY • **Add electrons revolving around the nucleus of an atom.**		
Step 3: Student-Led Activity (20 mins)	*The teacher guides the student to clone the code from Student Activity 1.* ***[Student Activity 1]*** ***Note**: The student will continue to add new functionality after teacher activity.*	
	Now we have the nucleus of the atoms representing the element; we need to show electrons revolving around the nucleus. How can we do that? Yes. We show the orbiting path of the electron. Remember what we had used to show the orbital path in the solar system?	**ESR**: We use the <a-sphere> entity and add animation to it. **ESR**: We used the <a-torus> entity. **ESR**: The for loop.

	Yes, we can use that here too. Let's do that. Since we want to do it for all the electrons for that element. What should we use to do that? Great! Also, we would want to show each electron revolving in a different orbit. Any ideas, how can we achieve that? We'll keep changing the rotation angle of the orbit. *Guide the student to add the entity to create an orbit path.* To add the **orbit** (torus to show the orbital path of revolution around the nucleus) entity:	**ESR**: Varied.

	• Create an 'a-entity' element using **document.createElement()**. • Set the **id**, **geometry**, **material**, **position** and **rotation** attribute using **.setAttribute()**. • Increase the orbit angle by 45 degrees to keep the orbital paths different for each electron. • Append the entity to the atom using **.appendChild()**.	

```
var orbitAngle = -180;

for (var num = 1; num <= numOfElectron; num++) {
  var orbit = document.createElement("a-entity");
  orbit.setAttribute("geometry", {
    primitive: "torus",
    arc: 360,
    radius: 0.28,
    radiusTubular: 0.001
  });

  orbit.setAttribute("material", {
    color: "#ff9e80",
    opacity: 0.3
  });

  orbit.setAttribute("position", {
    x: 0,
    y: 1,
    z: 0
  });

  orbit.setAttribute("rotation", {
    x: 0,
    y: orbitAngle,
    z: 0
  });

  orbitAngle += 45;

  atom.appendChild(orbit);
}
```

	Now we have the orbital paths for the electrons; we can add electrons and add animation for revolution. Do you remember how we do that?	**ESR**: For revolution, • We can create <a-entity> with animation components attached to it. • Then we can create <a-sphere> with some position to set the revolution radius.

	Guide the student to create the revolving electron. To add the **electron** (revolving around nucleus) entity: • Create an 'a-entity' element using **document.createElement()**. • Set the **id**, **rotation** and **animation** attribute using **.setAttribute()**. • Increase the angle by 65 degrees inside the loop to electrons moving in along the path. • Take a variable for the angle outside the loop:	

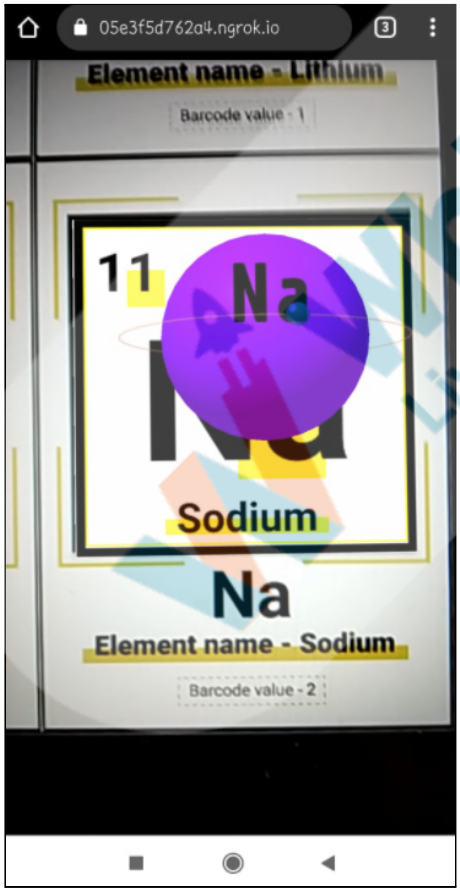
	`var electronAngle = 3` • Append the entity to the orbit using **.appendChild()**. • Create an 'a-entity' element using **document.createElement()**. • Set the **id**, **geometry**, **position** and **material** attribute using **.setAttribute()**. • Append the entity using **.appendChild()**.	

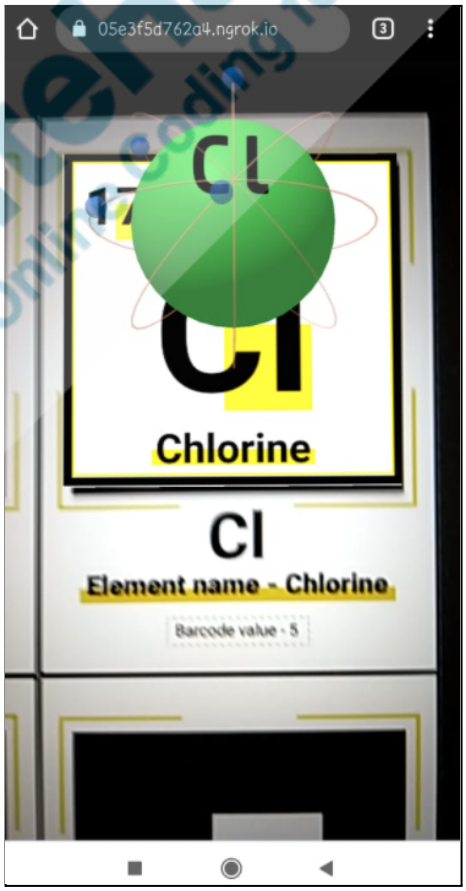
```
//<a-entity>
var electronGroup = document.createElement("a-entity");
electronGroup.setAttribute("id", `electron-group-${elementName}`);

electronGroup.setAttribute("rotation", {
    x: 0,
    y: 0,
    z: electronAngle
});

electronAngle += 65;

//animation
electronGroup.setAttribute("animation", {
    property: "rotation",
    to: `0 0 -360`,
    loop: "true",
    dur: 3500,
    easing: "linear"
});

orbit.appendChild(electronGroup);
```

```
//electron
var electron = document.createElement("a-entity");
electron.setAttribute("id", `electron-${elementName}`);
electron.setAttribute("geometry", {
    primitive: "sphere",
    radius: 0.02
});

electron.setAttribute("material", { color: "#0d47a1", opacity: 0.6 });

electron.setAttribute("position", {
    x: 0.2,
    y: 0.2,
    z: 0
});

electronGroup.appendChild(electron);
```

	Guide the student to test the output using ngrok.	

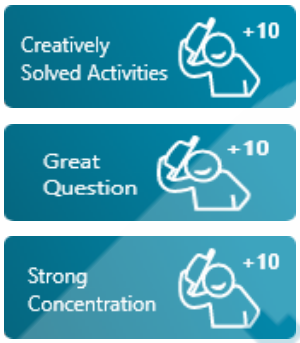
	***Note 1**: It would be better to take the printout of these marker images to see the output better.* ***Note 2**: Encourage the student to take the printout of the marker images and keep them ready each by cutting along the borders.*	





Teacher Guides Student to Stop Screen Share

WRAP UP SESSION - 5 mins
Teacher Starts Slideshow **Slide 21 to 25** 
Activity details **Following are the WRAP-UP session deliverables:** ● Appreciate the student. ● Revise the current class activities. ● Discuss the quizzes.
WRAP-UP QUIZ Click on In-Class Quiz
Continue WRAP-UP Session **Slide 26 to 31** 
Activity Details **Following are the session deliverables:** ● Explain the facts and trivia ● Next class challenge ● Project for the day ● Additional Activity (Optional)
FEEDBACK ● **Compliment the student for her/his effort in the class.** ● **Encourage the student to think and come up with their own solutions.**

You get a “hats-off”. Alright. See you in the next class.	*Make sure you have given at least 2 Hats Off during the class for:* 
PROJECT OVERVIEW DISCUSSION Refer the document below in Activity Links Sections	
Teacher Clicks 	
Additional Activities	*Encourage the student to write reflection notes in their reflection journal using markdown.* Use these as guiding questions: ● What happened today? ○ Describe what happened. ○ The code I wrote.
	The student uses the markdown editor to write their reflections in a reflection journal.

	• How did I feel after the class? • What have I learned about programming and developing games? • What aspects of the class helped me? What did I find difficult?	

Activity	Activity Name	Links
Teacher Activity 1	Boilerplate Code	https://github.com/whitehatjr/PRO-C174-Teacher-Boilerplate
Teacher Activity 2	3X3 Barcode Markers	https://github.com/artoolkit/ARToolKit5/tree/master/doc/patterns/Matrix%20code%203x3%20with%20parity%20(72dpi)
Teacher Activity 3	Teacher Reference Code	https://github.com/whitehatjr/PRO-C174
Student Activity 1	Boilerplate Code	https://github.com/whitehatjr/PRO-C174-Boilerplate
Teacher Reference 1	Ngrok Updates	https://docs.google.com/document/d/1dlMry188llEJl6rHEc3AkBashQSOwGQ40HQft29S8vQ/edit?usp=sharing
Teacher Reference 2	Project Document	https://s3-whjr-curriculum-uploads.whjr.online/70ee99ad-1d95-42d7-a25b-bbe065e5f976.pdf
Teacher	Project Solution	https://github.com/whitehatjr/PRO-C174-A

Reference 3		R
Teacher Reference 4	Visual-Aid	https://s3-whjr-curriculum-uploads.whjr.online/eb53650d-571e-4ebc-9737-47a8a80591cb.html
Teacher Reference 5	In-Class Quiz	https://s3-whjr-curriculum-uploads.whjr.online/dbcea75b-7d15-4fd1-8eef-968461b4baaf.pdf