University of Sunderland

Faculty: B.Sc. (Hons) Computer Systems Engineering

NAME: KRITAM CHAUDHARY

STUDENT NUMBER: 250773543

MODULE CODE: CET138

MODULE TITLE: Full Stack Development

MODULE LEADER: Subash Pariyar

SUBMISSION DATE AND TIME: 19th June 2026 at 23:59

ASSIGNMENT: 01

Academic Misconduct is an offence under university regulations, and this involves:

- **Plagiarism** – where you use information from another information source (including your previously submitted work) and pass it off as your own. This can be through direct copying, poor paraphrasing and/or absence of citations.
- **Collusion** – where you work too closely, intentionally, or unintentionally, with others to produce work that is similar in nature. This can be through loaning of materials, drafts or through unauthorised use of a fellow student's work.
- **Asking another person to write your assignment** – where you ask another individual or company to complete your work for you, be that paid or unpaid, and submit it as if it were your own.
- **Unauthorised use of artificial intelligence** – where you use artificial intelligence tools to generate your assignment instead of completing it yourself and/or where you have not been given permission to use artificial intelligence tools by your module leader. Please complete the following declaration around you use of artificial intelligence tools in your assignment.

STATEMENT ON USE OF ARTIFICIAL INTELLIGENCE TOOLS:

• I have used artificial intelligence tools to generate an idea for my assignment:	**NO**
• I have used artificial intelligence tools to write my assignment for me:	**NO**
• I have used artificial intelligence tools to brainstorm ideas for my assignment:	**NO**
• I have used artificial intelligence tools to correct my original assignment:	**NO**

DECLARATION

- I understand that by submitting this piece of work I am declaring it to be my own work and in compliance with the university regulations on Academic Integrity.
- I confirm that I have done this work myself without external support or inappropriate use of resources.
- I understand that I am only permitted to use artificial intelligence tools in line with guidance provided by my Module Leader, and I have not used artificial intelligence tools outside this remit.
- I confirm that this piece of work has not been submitted for any other assignment at this or another institution prior to this point in time.
- I can confirm that all sources of information, including quotations, have been acknowledged by citing the source in the text, along with producing a full list of the sources used at the end of the assignment.
- I understand that academic misconduct is an offence and can result in formal disciplinary proceedings.
- I understand that by submitting this assignment, I declare myself fit to be able to complete the assignment and I accept the outcome of the assessment as valid and appropriate.

Table of Contents

1. WHAT IS FULL STACK DEVELOPMENT?

Definition:

A full stack developer is someone who can build an entire website from start to finish.

"Full stack" means they know both the front-end (what users see) and the back end (where data is stored).

Two Parts:

1. FRONT-END (Client-Side)

- What the user sees in their browser
- Built with: HTML, CSS, JavaScript
- Example: When you type in a form, the form appears on your screen (front-end)
- Happens on: Your computer, in your browser

2. BACK-END (Server-Side)

- Where information is stored
- Built with: PHP, Python, Node.js, Java
- Example: When you submit a form, the data goes to a server and is saved in a database
- Happens on: Company's server, in the cloud

How They Work Together:

- User fills out a form on the website (FRONT-END)
- User clicks "Submit" button (FRONT-END event)
- JavaScript sends the data to the server (COMMUNICATION)
- Server receives it, checks if it's correct (BACK-END logic)
- Server saves it in database (BACK-END storage)
- Server sends back a "Success!" message
- User sees "Success!" on their screen (FRONT-END display)

For Agriculture:

- A farmer uses a website to register and buy fertilizer (FRONT-END)
- The website stores their order in a database (BACK-END)
- The company can track and deliver their order (BACK-END process)
- The farmer gets an email confirmation (BACK-END to FRONT-END)

Why Full Stack Skills Matter:

- You can build complete solutions
- You understand how everything works together
- You can fix problems anywhere in the application
- You're more valuable as a developer

2. HTML - HYPERTEXT MARKUP LANGUAGE

What is HTML?

HTML is the skeleton/structure of a website. It's like the blueprint of a house.

It tells the browser: "This is a heading, this is a paragraph, this is a button."

Why is it Important?

- Gives meaning to content (semantic HTML)
- Makes websites accessible to screen readers
- Helps search engines understand your page
- Without HTML, there's no webpage

Key HTML Elements We Used:

1. <form> - For collecting user input
2. <input> - For text, email, number, date fields
3. <label> - To label form fields (accessible)
4. <select> - For dropdown menus
5. <button> - For clickable buttons
6. <fieldset> & <legend> - To group related form fields

Our Example: Farmer Registration Form

What It Does:

- Farmers can enter their name, email, phone number
- They select their land area in acres
- They choose what type of fertilizer they want
- They pick a delivery date
- They agree to terms and submit

HTML Features Used:

- type="email" - Browser automatically validates email format
- type="tel" - Shows phone number keyboard on mobile
- type="date" - Shows calendar picker
- type="number" - Only allows numbers
- required - Makes field mandatory
- <label for="fieldname"> - Connects label to input

Accessibility:

- Screen readers can read form labels
- Users know what each field is for
- Keyboard users can navigate easily
- Form validates before submission

Why This Matters:

HTML gives structure and meaning. Without proper HTML, your website isn't accessible to everyone.

It's the foundation - CSS and JavaScript build on top of it.

3. CSS - CASCADING STYLE SHEETS

What is CSS?

CSS makes websites look beautiful. If HTML is the skeleton, CSS is the clothes and makeup.

It controls colors, fonts, spacing, animations, and layouts.

Why is it Important?

- Makes websites attractive
- Creates responsive designs (works on all devices)
- Improves user experience
- Separates content (HTML) from styling (CSS)

CSS Concepts We Used:

1. COLORS & GRADIENTS

background: linear-gradient(135deg, #6366f1, #ec4899);

- Creates a smooth color transition
- Makes cards look modern

2. HOVER EFFECTS

.css-skill-card:hover {

transform: translateY(-8px);

box-shadow: 0 15px 30px rgba(0,0,0,0.15);}

- When mouse hovers over card, it lifts up
- Shadow appears underneath
- Makes it interactive and fun

3. TRANSITIONS & ANIMATIONS

transition: all 0.4s cubic-bezier(0.4, 0, 0.2, 1);

- Smooth movement from one state to another
- Feels natural and professional

4. CSS GRID

display: grid;

grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));

- Automatically arranges cards in rows
- Responsive - adapts to screen size
- On desktop: 3 cards per row
- On mobile: 1 card per row

5. RESPONSIVE DESIGN

- Uses media queries to adapt to different screens
- Same content, different layouts
- Works on phones, tablets, desktops

Our Example: Farming Skills Cards

What We Did:

1. Created boxes with borders
2. Added icons inside boxes
3. Added hover animation (lift up + shadow)
4. Icon scales larger on hover
5. Cards arranged in responsive grid
6. Colors match agriculture theme (green for farming, blue for water, etc.)

CSS Properties Used:

- border-radius: Makes corners rounded
- box-shadow: Adds depth
- transform: Moves/scales elements
- opacity: Controls transparency
- z-index: Controls layering

Why This Matters:

CSS makes websites look professional and modern.

It also makes websites work on all devices (responsive design).

Good CSS = Happy users

4. BOOTSTRAP FRAMEWORK

What is Bootstrap?

Bootstrap is a pre-built CSS framework that gives you ready-made components.

It's like LEGO for websites - you use pre-made pieces instead of building everything from scratch.

Why Use Bootstrap?

- Saves time - don't write CSS from scratch
- Responsive by default - works on all devices
- Professional look - built-in modern design
- Consistent styling - same look across components
- Easy to use - just add class names

Bootstrap Grid System:

The container:

- Bootstrap divides screens into 12 columns
- You can make components take 1-12 columns
- On desktop: col-md-6 means "take 6 columns (half width)"
- On mobile: Full width automatically

Example:

 Box 1

 Box 2

On desktop: 2 boxes side by side (each 50% width)

On mobile: 2 boxes stacked vertically (each 100% width)

Our Example: Farm Management Dashboard

What We Built:

- 4 stat boxes showing farm metrics

- Box 1: Total Yield (1,248 kg)

- Box 2: Fertilizer Used (342 L)

- Box 3: Water Usage (5,600 L)

- Box 4: Pest Issues (2)

Bootstrap Classes We Used:

- col-md-6 - Makes boxes take half width on medium screens

- h-100 - Makes all boxes same height

- border-left - Colored left border on each box

- text-muted - Gray text for labels

- fw-bold - Bold font weight

Responsive Behavior:

On Desktop: 2x2 grid (2 boxes per row)

On Tablet: 2x2 or 1x4 (depends on tablet size)

On Mobile: 4x1 stacked (1 box per row)

The Magic:

Bootstrap automatically handles all this with just class names.

You don't write any CSS - it's already included.

Just add class="col-md-6" and it handles the responsive part.

Why This Matters:

Bootstrap lets developers build websites 10x faster.

It's industry standard - most companies use it.

Great for building MVPs (Minimum Viable Products) quickly.

5. JAVASCRIPT - CLIENT-SIDE PROGRAMMING

What is JavaScript?

JavaScript makes websites interactive. It runs in your browser and responds to user actions.

Without JavaScript, websites are just static text and images.

Why is it Important?

- Makes websites interactive and responsive
- No page reload needed (smooth experience)
- Validates form data before sending
- Creates animations and effects
- Powers modern web applications

JavaScript Basics We Used:

1. VARIABLES

```
let landArea = 10;
let yieldPerAcre = 250;
```

- Stores information in memory
- Can be used later

2. DOCUMENT MANIPULATION

document.getElementById('jsTotal')

- Finds elements in the HTML
- Allows us to change them

3. EVENT LISTENERS

oninput="calculateYield()"

- Triggers when user types in a field
- Calls the function immediately

4. CALCULATIONS

totalYield = landArea * yieldPerAcre;

- Does math based on user input
- Updates result in real-time

5. UPDATING DOM

element.innerText = totalYield;

- Changes text on the page
- No page refresh needed

Our Example: Crop Yield Estimator

How It Works:

1. User enters land area (e.g., 10 acres)
2. User enters yield per acre (e.g., 250 kg)
3. User selects crop type (e.g., rice)

JavaScript then:

1. Grabs the values from input fields

2. Multiplies them: 10 * 250 = 2500

3. Updates the result on the page instantly

Code Breakdown:

```javascript
function calculateYield() {
  // Get values from input fields
  let landArea = parseFloat(document.getElementById('jsLandArea').value);
  let yieldPerAcre = parseFloat(document.getElementById('jsYieldPerAcre').value);

  // Calculate total yield
  let totalYield = landArea * yieldPerAcre;

  // Update the page
  document.getElementById('jsTotal').innerText = totalYield.toFixed(0);
}
```

6. CONCLUSION

What I Learned

Building this agriculture-themed portfolio has been an incredible learning journey. I started by understanding that web development isn't just about writing code, it's about creating complete, working solutions that solve real problems. Through this assignment, I discovered that a "full stack developer" is someone who can see the big picture: from how a farmer interacts with a website (front-end) to how their data is safely stored and processed on servers (back-end). Each technology I learned HTML, CSS, Bootstrap, and JavaScript plays a specific role in this ecosystem. HTML

gave structure and meaning to my content, CSS made it beautiful and responsive, Bootstrap saved me hours of repetitive work, and JavaScript brought everything to life with real-time calculations. The agriculture theme kept me motivated because I could see practical applications: farmers registering for fertilizer, tracking their crop yields, and managing their farms through a professional, modern website.

Why This Matters & Moving Forward

What excites me most is realizing that these skills aren't just theoretical, they're immediately applicable to real-world problems. Whether it's building an e-commerce site for agricultural products, creating a dashboard for farmers to track their yields, or developing a registration system for fertilizer sales, I now have the foundational knowledge to build these solutions. I understand that responsive design means my website works on a farmer's phone just as well as on a desktop computer. I know that proper HTML structure and CSS styling make websites accessible to everyone, including people using screen readers. I've learned that Bootstrap isn't "cheating" it's how professional developers work efficiently. Moving forward, I'm excited to deepen my skills by learning backend technologies like PHP or Python, database management, and how to connect everything together. This assignment has shown me that full stack development is a continuous journey of learning, but with these fundamentals in place, I'm confident I can build meaningful web applications that make a difference.