WAD621S – Lab 4 Assignment Brief

Title: Registration → Profile Cards (HTML/CSS + JS DOM + Tables Forms)

Duration: 25 August - 2 September (You can it in a day)
Weight: 30 marks (continuous assessment)

Objectives

- Build an accessible registration form using semantic HTML and CSS.
- Implement inline validation and user feedback using JavaScript.
- Dynamically render a profile card for each submission.
- Maintain a summary table synchronized with the rendered cards.

Important Notice

Lab 4 is STRICTLY INDIVIDUAL WORK. Each student must submit their own repository and README. No group submissions will be accepted. Collaboration is reserved for the final project, not this lab.

Tasks

1. Complete the registration form (First/Last Name, Email, Programme, Year, Interests, Photo URL).
2. Validate inputs (required fields, email format, year selection).
3. On submit, create a profile card and add a row to the summary table.
4. Implement a Remove action that deletes both the card and the corresponding table row.

Accessibility

- Label all inputs; ensure keyboard navigation.
- Provide inline error messages and update an aria-live region for feedback.
- Use adequate colour contrast and readable font sizes.

Stretch Goals (optional)

- Add an Edit feature to update existing profiles.
- Persist profiles with LocalStorage and restore on page load.
- Add a search/filter input to the cards/table.

Rubric (30 marks)

Criterion	Marks	Notes
Form completeness + accessibility	6	Labels, correct input types, responsive, accessibility checks
Validation & error handling	6	Inline messages, prevents invalid submits, aria-live
Profile card creation	8	Dynamic DOM generation, includes remove button
Summary table integration	6	Accurate updates in sync with form submissions and removals
Code quality	4	Semantic HTML, clean CSS, readable JS, comments included

Submission

- Submit a ZIP containing `index.html`, `style.css`, `script.js`. Include a short README with your names and notes.
- Ensure the page runs locally in a browser without extra setup.

Appendix: Git Instructions & Sample Code

A. Git Workflow (Required for this Lab)

- Install Git: `https://git-scm.com/downloads`
- Set your identity (first time only):

```
git config --global user.name "Your Name"
git config --global user.email "you@example.com"
```

- Create repo in project folder:

```
git init
```

- Stage and commit changes frequently:

```
git add .
git commit -m "Initial form + layout"
```

- Create a new branch for features:

```
git checkout -b feature/cards
```

- Link to GitHub:

```
git remote add origin https://github.com/<user>/<repo>.git
```

- Push your branch:

```
git push -u origin feature/cards
```

- Open a Pull Request (PR) on GitHub and review with teammates before merging.

Evidence to Submit (Git):

- GitHub repository URL in README.md
- At least 5 commits with meaningful messages
- Optional: screenshot of PR discussion/comments

B. Sample Code Snippets

1) Accessible HTML Form (excerpt)

```
<form id="regForm" novalidate>
  <label for="email">Email <span aria-hidden="true">*</span></label>
  <input type="email" id="email" name="email" required/>
  <small class="error" id="err-email"></small>

  <fieldset>
    <legend>Year of Study <span aria-hidden="true">*</span></legend>
    <label><input type="radio" name="year" value="1" required/> 1</label>
    <label><input type="radio" name="year" value="2"/> 2</label>
  </fieldset>
  <div aria-live="polite" class="sr" id="live"></div>
  <button type="submit">Add Student</button>
</form>
```

2) JavaScript: Basic Validation

```
function validateEmail(value){
  const ok = /^[^\s@]+@[^\s@]+\.[^\s@]+$/.test(value);
  const err = document.getElementById("err-email");
  if(!ok){ err.textContent = "Please enter a valid email."; }
  else { err.textContent = ""; }
  return ok;
}

document.getElementById("regForm").addEventListener("submit", (e) => {
  const value = document.getElementById("email").value;
  if(!validateEmail(value)){
    e.preventDefault();
    document.getElementById("live").textContent = "Fix errors before submitting.";
  }
});
```

3) JavaScript: Create Card + Table Row (excerpt)

```
function addEntry(data){
  // Card
  const card = document.createElement("div"); card.className = "card-person";
  card.innerHTML = `<img src="${data.photo||"https://placehold.co/128"}" alt="">
    <div><h3>${data.first} ${data.last}</h3>
    <p><span class="badge">${data.prog}</span> <span class="badge">Year ${data.year}<
  document.getElementById("cards").prepend(card);

  // Table
  const tr = document.createElement("tr");
  tr.innerHTML = `<td>${data.first} ${data.last}</td><td>${data.prog}</td><td>${data.
  document.querySelector("#summary tbody").prepend(tr);
}
```

C. Resources & Links

- MDN Web Docs (HTML/JS/CSS): `https://developer.mozilla.org/`
- MDN Forms Guide: `https://developer.mozilla.org/en-US/docs/Learn/Forms`
- MDN Accessibility: `https://developer.mozilla.org/en-US/docs/Web/Accessibility`
- WAVE Accessibility Tool: `https://wave.webaim.org/`
- Lighthouse (Chrome DevTools): Inspect → Lighthouse
- Git official docs: `https://git-scm.com/docs`
- GitHub Guides: `https://guides.github.com/`
- Atlassian Git Tutorials: `https://www.atlassian.com/git`
- Chart.js: `https://www.chartjs.org/`
- Leaflet.js: `https://leafletjs.com/`
- EmailJS: `https://www.emailjs.com/`