Cypress for QA Problem Set 3 Solutions

Useful examples can be found here: https://github.com/cypress-io/cypress-example-recipes

1 Feature Flags and Config-Driven UI

Problem 1.1: Respect feature flag and render new button only if enabled

```
it('should show Beta button only if flag is enabled', ()
    => {
    cy.intercept('GET', '/api/feature-flags', {
        body: { showBeta: true }
    }).as('getFlags');

    cy.visit('/');
    cy.wait('@getFlags');
    cy.get('#beta-feature-btn').should('exist');
});
```

Problem 1.2: Assert feature flag disables UI elements

```
it('should hide premium section if user lacks premium
   flag', () => {
    cy.intercept('GET', '/api/user-profile', {
        body: { name: 'Jane', flags: { premium: false }
           }
    }).as('getProfile');

    cy.visit('/dashboard');
```

```
    cy.wait('@getProfile');
    cy.get('#premium-section').should('not.exist');
});
```

2 Time Manipulation and Scheduled Behavior

Problem 2.1: Freeze clock and simulate time-dependent popup

```
it('should show popup 10 seconds after load', () => {
    cy.clock();
    cy.visit('/');
    cy.tick(10000);
    cy.get('#timed-popup').should('be.visible');
});
```

Problem 2.2: Test expiry banner for outdated sessions

```
it('should show session expired banner after timeout',
   () => {
    cy.clock();
    cy.visit('/profile');
    cy.tick(1800000); // 30 mins
    cy.get('.session-expired').should('be.visible');
});
```

3 File Uploads and Downloads

Problem 3.1: Upload a file and confirm success

```
it('should upload file and return confirmation', () => {
    const fileName = 'test-data.csv';
    cy.get('input[type=file]').selectFile(`cypress/
       fixtures/${fileName}`);
    cy.get('#upload-btn').click();
```

```
    cy.get('.upload-success').should('contain', 'Upload
       complete');
});
```

Problem 3.2: Check if thhe downloaded file contains the expected content

```
it('should download file with correct content', () => {
    cy.intercept('GET', '/api/report', {
        headers: { 'content-disposition': 'attachment;
           filename="report.csv"' },
        body: 'name,score\nAlice,90\nBob,85'
    }).as('getReport');

    cy.get('#download-report').click();
    cy.wait('@getReport').its('response.body').should('
       include', 'Alice');
});
```

4 Session & LocalStorage Validation

Problem 4.1: Ensure session cookie is set post login

```
it('should store auth cookie on login', () => {
    cy.visit('/login');
    cy.get('#email').type('test@example.com');
    cy.get('#password').type('123456');
    cy.get('#login-btn').click();

    cy.getCookie('auth_token').should('exist');
});
```

Problem 4.2: Persist localStorage and reuse across tests

```
beforeEach(() => {
    cy.restoreLocalStorage();
});

afterEach(() => {
    cy.saveLocalStorage();
});

it('should store and reuse token from localStorage', ()
   => {
    cy.setLocalStorage('token', 'abc123');
    cy.reload();
    cy.getLocalStorage('token').should('eq', 'abc123');
});
```

5 Network Errors and Resilience Testing

Problem 5.1: Simulate 500 error and assert UI fallback

```
it('should show error banner on 500 failure', () => {
    cy.intercept('GET', '/api/data', { statusCode: 500
       }).as('getData');
    cy.visit('/analytics');
    cy.wait('@getData');
    cy.get('.error-banner').should('contain', 'Unable to
        load data');
});
```

Problem 5.2: Retry logic on fetch failure

```
it('should retry failed fetch up to 3 times', () => {
    let attempt = 0;

    cy.intercept('GET', '/api/fetch-metrics', (req) => {
        attempt++;
        req.reply(attempt < 3 ? { statusCode: 502 } : {
           statusCode: 200, body: { success: true } });
```

```
    }).as('fetchMetrics');

    cy.visit('/metrics');
    cy.wait('@fetchMetrics');
    cy.get('.metrics-status').should('contain', 'Data
       loaded');
});
```

Problem 5.3: Simulate delayed API response and assert loading indicator

```
it('should show loader while data is being fetched', ()
   => {
    cy.intercept('GET', '/api/stats', (req) => {
        req.on('response', (res) => {
            res.setDelay(3000); // 3 second delay
        });
        req.reply({ body: { views: 120 } });
    }).as('getStats');

    cy.visit('/stats');
    cy.get('.loading-spinner').should('be.visible');
    cy.wait('@getStats');
    cy.get('.loading-spinner').should('not.exist');
});
```

Problem 5.4: Retry form submission after initial failure

```
it('should retry form submission on failure and
   eventually succeed', () => {
    let callCount = 0;

    cy.intercept('POST', '/api/submit-form', (req) => {
        callCount++;
        req.reply(callCount === 1 ? { statusCode: 503 }
           : { statusCode: 200 });
    }).as('submitForm');
```

```
    cy.get('#form-input').type('Some data');
    cy.get('#submit-btn').click();
    cy.wait('@submitForm');
    cy.get('.status-message').should('contain', '
       Submission successful');
});
```

Problem 5.5: Disable retry logic and ensure that failure is shown

```
it('should fail fast when retry is disabled', () => {
    Cypress.config('defaultCommandTimeout', 3000); //
       Set short timeout

    cy.intercept('GET', '/api/slow-data', {
        delay: 10000,
        statusCode: 200,
        body: { data: [] }
    }).as('slowRequest');

    cy.visit('/slow-page');
    cy.wait('@slowRequest', { timeout: 4000 }).its('
       response.statusCode').should('eq', 200);
    cy.get('.timeout-warning').should('be.visible');
});
```

6 Visual Regression (Setup Placeholder)

Problem 6.1: Capture snapshot for visual testing (using plugin)

```
// Assuming use of cypress-image-snapshot
it('should match UI snapshot', () => {
    cy.visit('/dashboard');
    cy.matchImageSnapshot('dashboard-ui');
});
```

Problem 6.2: Compare the before-and-after state of the modal window

```
it('should match snapshot after opening modal', () => {
    cy.visit('/products');
    cy.get('#view-details').click();
    cy.get('.modal-content').should('be.visible');
    cy.matchImageSnapshot('modal-open');
});
```

Problem 6.3: Ensure that the theme switch does not break the layout

```
it('should match layout snapshot after switching theme',
    () => {
    cy.visit('/profile');
    cy.get('#theme-toggle').click();
    cy.get('body').should('have.class', 'dark-mode');
    cy.matchImageSnapshot('dark-mode-profile');
});
```

Problem 6.4: Visual diff with masked dynamic content

```
it('should ignore dynamic timestamp in visual diff', ()
   => {
    cy.visit('/dashboard');
    cy.get('.dynamic-timestamp').invoke('hide'); // Hide
        dynamic content
    cy.matchImageSnapshot('dashboard-static');
});
```