**Seminarul 7 — Partea 3: Proiecte/teme (extins)**

**Metodologie & obiective**

Partea 3 livrează \*\*45 de proiecte\*\* axate pe \*\*Fetch API, formulare, JSON și persistență minimă\*\*. Fiecare proiect include: \*\*Scop didactic\*\*, \*\*Specificații\*\*, \*\*Criterii de acceptare\*\* (măsurabile), \*\*Soluție (rezumat)\*\* și \*\*AI‑assist (VSL)\*\*. Arhivele conțin startere executabile (mock Express, structuri modulare, teste unitare Vitest & Jest).

Strategia de consolidare: \*\*L1\*\* (fundamente) → \*\*L2\*\* (politici, validare extinsă, persistență) → \*\*L3\*\* (arhitecturi: queue offline, IndexedDB, circuit breaker, i18n, observability).

**Proiecte L1 (Fundamente) — 15 teme**

**L1 — P01. GET /api/clubs — listare de bază (map, error split 4xx/5xx)**

\*\*Scop didactic.\*\* Reflexe corecte în folosirea Fetch, formulare semantice și serializare JSON.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: jsonGet\\(, fetch\\(. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Construiește serviciu `jsonGet('/api/clubs')`, afișează listă, diferențiază 4xx/5xx.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L1 — P02. Formular semantic — `required`, `type=email`, `minlength`, aria‑live**

\*\*Scop didactic.\*\* Reflexe corecte în folosirea Fetch, formulare semantice și serializare JSON.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: FormData, required, type=\"email\". HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* HTML semantic + `FormData → object → validate → JSON` + focus management.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L1 — P03. FormData → JSON — serializare & normalizare**

\*\*Scop didactic.\*\* Reflexe corecte în folosirea Fetch, formulare semantice și serializare JSON.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: FormData, required, type=\"email\". HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* HTML semantic + `FormData → object → validate → JSON` + focus management.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L1 — P04. POST JSON — `Content-Type`/`Accept` corecte + success UI**

\*\*Scop didactic.\*\* Reflexe corecte în folosirea Fetch, formulare semantice și serializare JSON.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: jsonPost\\(, JSON\\.stringify. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Trimite `jsonPost('/api/registrations', data)` cu antete, tratează 201/422.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L1 — P05. Loading/empty state — UX prietenos**

\*\*Scop didactic.\*\* Reflexe corecte în folosirea Fetch, formulare semantice și serializare JSON.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L1 — P06. Defensive JSON parsing — `content-type` + try/catch**

\*\*Scop didactic.\*\* Reflexe corecte în folosirea Fetch, formulare semantice și serializare JSON.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L1 — P07. Reset pe succes + focus management**

\*\*Scop didactic.\*\* Reflexe corecte în folosirea Fetch, formulare semantice și serializare JSON.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L1 — P08. Mesaje de eroare „umanizate” (map HTTP)**

\*\*Scop didactic.\*\* Reflexe corecte în folosirea Fetch, formulare semantice și serializare JSON.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L1 — P09. Debounce input (300 ms) pentru cost mic**

\*\*Scop didactic.\*\* Reflexe corecte în folosirea Fetch, formulare semantice și serializare JSON.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: setTimeout. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L1 — P10. Retry (1x) pe network error; NU pe 4xx**

\*\*Scop didactic.\*\* Reflexe corecte în folosirea Fetch, formulare semantice și serializare JSON.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: retries, backoff. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Retry controlat (doar 5xx/rețea) cu backoff + jitter; fără retry pe 4xx.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L1 — P11. Select „club” populat din GET**

\*\*Scop didactic.\*\* Reflexe corecte în folosirea Fetch, formulare semantice și serializare JSON.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: jsonGet\\(, fetch\\(. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Construiește serviciu `jsonGet('/api/clubs')`, afișează listă, diferențiază 4xx/5xx.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L1 — P12. Buton `disabled` în `submitting`**

\*\*Scop didactic.\*\* Reflexe corecte în folosirea Fetch, formulare semantice și serializare JSON.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L1 — P13. Mic logger non‑PII (tip + durată)**

\*\*Scop didactic.\*\* Reflexe corecte în folosirea Fetch, formulare semantice și serializare JSON.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include cel puțin unul dintre: console\\.log, performance\\.now. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L1 — P14. Headers consistente — `Accept`/`Content-Type`**

\*\*Scop didactic.\*\* Reflexe corecte în folosirea Fetch, formulare semantice și serializare JSON.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L1 — P15. Arhitectură modulară (api/forms/services/utils)**

\*\*Scop didactic.\*\* Reflexe corecte în folosirea Fetch, formulare semantice și serializare JSON.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: FormData, required, type=\"email\". HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* HTML semantic + `FormData → object → validate → JSON` + focus management.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**Proiecte L2 (Intermediare) — 15 teme**

**L2 — P01. `AbortController` + timeout (6–8 s)**

\*\*Scop didactic.\*\* Politici robuste (timeout, retry), validare extinsă, cache TTL + SWR, autosave și UX rezilient.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: AbortController. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* `AbortController` oprește cererea după 6–8 s; UI afișează mesaj clar.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L2 — P02. Retry(3) cu exponential backoff + jitter (5xx/rețea)**

\*\*Scop didactic.\*\* Politici robuste (timeout, retry), validare extinsă, cache TTL + SWR, autosave și UX rezilient.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: retries, backoff. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Retry controlat (doar 5xx/rețea) cu backoff + jitter; fără retry pe 4xx.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L2 — P03. Validare custom (regex email, lungimi, consent)**

\*\*Scop didactic.\*\* Politici robuste (timeout, retry), validare extinsă, cache TTL + SWR, autosave și UX rezilient.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: validateEnrol\\(. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L2 — P04. `aria-live` + focus pe primul câmp invalid**

\*\*Scop didactic.\*\* Politici robuste (timeout, retry), validare extinsă, cache TTL + SWR, autosave și UX rezilient.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L2 — P05. localStorage cache TTL (5 min) pentru GET**

\*\*Scop didactic.\*\* Politici robuste (timeout, retry), validare extinsă, cache TTL + SWR, autosave și UX rezilient.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: jsonGet\\(, fetch\\(, localStorage, lsSet\\(, lsGet\\(. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Construiește serviciu `jsonGet('/api/clubs')`, afișează listă, diferențiază 4xx/5xx. `lsGet/lsSet` cu TTL; afişezi cache imediat, reîmprospătezi în fundal.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L2 — P06. SWR minimal (livrezi cache, apoi refresh)**

\*\*Scop didactic.\*\* Politici robuste (timeout, retry), validare extinsă, cache TTL + SWR, autosave și UX rezilient.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: refreshClubs\\(, lsSet\\(, localStorage, lsSet\\(, lsGet\\(. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* `lsGet/lsSet` cu TTL; afişezi cache imediat, reîmprospătezi în fundal.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L2 — P07. Draft autosave + restore**

\*\*Scop didactic.\*\* Politici robuste (timeout, retry), validare extinsă, cache TTL + SWR, autosave și UX rezilient.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L2 — P08. Optimistic UI la POST + reconciliere**

\*\*Scop didactic.\*\* Politici robuste (timeout, retry), validare extinsă, cache TTL + SWR, autosave și UX rezilient.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: jsonPost\\(, JSON\\.stringify. JS include cel puțin unul dintre: status:\\s\*'success', updateUI\\(. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Trimite `jsonPost('/api/registrations', data)` cu antete, tratează 201/422. Adaugă în UI înscriere „optimistică”; la eroare, revoci și afișezi motivul.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L2 — P09. Error mapping (network/4xx/5xx → mesaje)**

\*\*Scop didactic.\*\* Politici robuste (timeout, retry), validare extinsă, cache TTL + SWR, autosave și UX rezilient.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L2 — P10. Idempotency guard local (nu dubla POST‑uri)**

\*\*Scop didactic.\*\* Politici robuste (timeout, retry), validare extinsă, cache TTL + SWR, autosave și UX rezilient.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: jsonPost\\(, JSON\\.stringify. JS include cel puțin unul dintre: idempotenc, retries. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Trimite `jsonPost('/api/registrations', data)` cu antete, tratează 201/422.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L2 — P11. Feature flags pentru retry/timeouts**

\*\*Scop didactic.\*\* Politici robuste (timeout, retry), validare extinsă, cache TTL + SWR, autosave și UX rezilient.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: retries, backoff, AbortController. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Retry controlat (doar 5xx/rețea) cu backoff + jitter; fără retry pe 4xx. `AbortController` oprește cererea după 6–8 s; UI afișează mesaj clar.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L2 — P12. Metrics minime (durată request)**

\*\*Scop didactic.\*\* Politici robuste (timeout, retry), validare extinsă, cache TTL + SWR, autosave și UX rezilient.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L2 — P13. Redirect policy (concept) & handling**

\*\*Scop didactic.\*\* Politici robuste (timeout, retry), validare extinsă, cache TTL + SWR, autosave și UX rezilient.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L2 — P14. CORS sanity (mock preflight)**

\*\*Scop didactic.\*\* Politici robuste (timeout, retry), validare extinsă, cache TTL + SWR, autosave și UX rezilient.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L2 — P15. Security hints: XSS safe outputs**

\*\*Scop didactic.\*\* Politici robuste (timeout, retry), validare extinsă, cache TTL + SWR, autosave și UX rezilient.

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**Proiecte L3 (Avansate) — 15 teme**

**L3 — P01. Fetch wrapper configurabil (baseURL, headers, interceptors)**

\*\*Scop didactic.\*\* Arhitecturi și concepte avansate (queue, IndexedDB, circuit breaker, i18n, observability).

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L3 — P02. Batching (coalesce read) — concept + demo**

\*\*Scop didactic.\*\* Arhitecturi și concepte avansate (queue, IndexedDB, circuit breaker, i18n, observability).

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L3 — P03. Offline queue pentru POST — concept + schelet**

\*\*Scop didactic.\*\* Arhitecturi și concepte avansate (queue, IndexedDB, circuit breaker, i18n, observability).

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: jsonPost\\(, JSON\\.stringify. JS include cel puțin unul dintre: queue, offline. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Trimite `jsonPost('/api/registrations', data)` cu antete, tratează 201/422. Coadă locală pentru `POST` când offline; sincronizare la `online` (concept & schelet).

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L3 — P04. IndexedDB mini‑cache (read‑through) — schelet**

\*\*Scop didactic.\*\* Arhitecturi și concepte avansate (queue, IndexedDB, circuit breaker, i18n, observability).

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: localStorage, lsSet\\(, lsGet\\(. JS include cel puțin unul dintre: indexedDB. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* `lsGet/lsSet` cu TTL; afişezi cache imediat, reîmprospătezi în fundal. Deschide DB, creează store; păstrezi răspunsuri mari; fallback la localStorage.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L3 — P05. Prioritizare requests (low/high) — concept**

\*\*Scop didactic.\*\* Arhitecturi și concepte avansate (queue, IndexedDB, circuit breaker, i18n, observability).

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L3 — P06. Circuit breaker simplificat — schelet**

\*\*Scop didactic.\*\* Arhitecturi și concepte avansate (queue, IndexedDB, circuit breaker, i18n, observability).

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L3 — P07. ETag/If‑None‑Match — simulare în mock**

\*\*Scop didactic.\*\* Arhitecturi și concepte avansate (queue, IndexedDB, circuit breaker, i18n, observability).

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include cel puțin unul dintre: If-None-Match, etag. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Include `If-None-Match` la GET; tratează `304 Not Modified` (simulat în mock).

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L3 — P08. Pagination + caching per pagină**

\*\*Scop didactic.\*\* Arhitecturi și concepte avansate (queue, IndexedDB, circuit breaker, i18n, observability).

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include cel puțin unul dintre: page, pagination. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Chei de cache per pagină; parametru `?page=`; UI pentru pager.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L3 — P09. Search cu debounce + cancel (AbortController)**

\*\*Scop didactic.\*\* Arhitecturi și concepte avansate (queue, IndexedDB, circuit breaker, i18n, observability).

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: setTimeout, AbortController. JS include cel puțin unul dintre: AbortController, setTimeout. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* `AbortController` oprește cererea după 6–8 s; UI afișează mesaj clar. Debounce + cancel (AbortController) la căutare succesivă.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L3 — P10. Schema validation (predicate stricte)**

\*\*Scop didactic.\*\* Arhitecturi și concepte avansate (queue, IndexedDB, circuit breaker, i18n, observability).

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include: validateEnrol\\(. JS include cel puțin unul dintre: typeof, Array\\.isArray. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L3 — P11. Error boundary UI (fallback state)**

\*\*Scop didactic.\*\* Arhitecturi și concepte avansate (queue, IndexedDB, circuit breaker, i18n, observability).

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L3 — P12. Bulk submit cu „transaction‑like” local**

\*\*Scop didactic.\*\* Arhitecturi și concepte avansate (queue, IndexedDB, circuit breaker, i18n, observability).

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L3 — P13. Feature flags A/B pe policy fetch**

\*\*Scop didactic.\*\* Arhitecturi și concepte avansate (queue, IndexedDB, circuit breaker, i18n, observability).

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L3 — P14. I18n pentru mesaje de eroare**

\*\*Scop didactic.\*\* Arhitecturi și concepte avansate (queue, IndexedDB, circuit breaker, i18n, observability).

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include cel puțin unul dintre: i18n, locale. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**L3 — P15. Observability light: logger + tags**

\*\*Scop didactic.\*\* Arhitecturi și concepte avansate (queue, IndexedDB, circuit breaker, i18n, observability).

\*\*Specificații.\*\* Aplică modelul semantic HTML, trimite/primește \*\*JSON\*\* prin Fetch, tratează explicit erorile (network/HTTP), păstrează `Content‑Type/Accept` corecte și separă responsabilitățile (api/forms/services/utils). Pentru L2 adaugă `AbortController`, `retry`, cache TTL + SWR, autosave; pentru L3 folosește scheletele din `src/addons/advanced.js` ca punct de plecare.

\*\*Criterii de acceptare.\*\* JS include cel puțin unul dintre: console\\.log, performance\\.now. HTML include: <form id=\"enrol\", aria-live.

\*\*Soluție (rezumat).\*\* Aplică principiile generale: separare, validare, erori mapate, accesibilitate.

\*\*AI‑assist (VSL).\*\* Exemplu: „Generează `retryPolicy` cu backoff+jitter”, „Scrie teste pentru `validateEnrol` (cazuri limită)”, „Propune chei/TTL pentru cache + SWR”.

**Rubrică (evaluare)**

Funcționalitate & UX (40%), Calitatea codului (30%), Accesibilitate (15%), Testare (15%).

**Referințe (APA 7, cu DOI)**

Fielding, R., Nottingham, M., & Reschke, J. (2022). HTTP Semantics (RFC 9110). RFC Editor. https://doi.org/10.17487/RFC9110

Bray, T. (2017). The JavaScript Object Notation (JSON) Data Interchange Format (RFC 8259). RFC Editor. https://doi.org/10.17487/RFC8259

Nottingham, M., & Wilde, E. (2016). Problem Details for HTTP APIs (RFC 7807). RFC Editor. https://doi.org/10.17487/RFC7807

Klyne, G., & Newman, C. (2002). Date and Time on the Internet: Timestamps (RFC 3339). RFC Editor. https://doi.org/10.17487/RFC3339

Barth, A. (2011). The Web Origin Concept (RFC 6454). RFC Editor. https://doi.org/10.17487/RFC6454