Här är ett utkast till en uppdaterad teknisk specifikation, version 1.2. Jag har försökt inkorporera de ändringar och förtydliganden vi diskuterat. Jag har också lagt till några punkter under "Potentiella framtida utökningar" som kan vara värda att fundera på längre fram.

Läs igenom den noggrant och se om den stämmer överens med din nuvarande vision och de beslut vi tagit.

**Teknisk specifikation: Webbapplikation för digital tillsyn  
Version: 1.2 (Uppdaterad med förtydliganden kring UI-flöden, knapptexter, i18n-struktur, och checkbox-hantering)  
Datum: 2025-05-14 (Exempeldatum)**

**1. Övergripande Mål och Syfte**  
Webbapplikationen ska möjliggöra för användare att genomföra digital tillsyn av webbsidor, appvyer eller andra digitala enheter baserat på en uppladdad regelfil i JSON-format. Appen ska guida användaren genom processen att definiera stickprov, granska krav, dokumentera observationer, spara/ladda pågående arbete och exportera resultat.

**2. Teknisk Stack & Arkitektur**

* **Frontend:** HTML5, CSS3, ren JavaScript (ES6+).
* **Backend:** Ingen databaskoppling i denna version. Appen körs lokalt (t.ex. via IIS på Windows 11, eller annan enkel webbserver).
* **Körningsmiljö:** Primärt för skrivbordswebbläsare (moderna versioner av Chrome, Firefox, Edge).
* **Kodstruktur:** Modulär och komponentbaserad.
  + **JavaScript-filer (Logik/Hjälp):** Placeras i js/ (t.ex. main.js, state.js, utils/helpers.js, i18n/sv-SE.json). Dessa ska vara IIFE:er (Immediately Invoked Function Expressions) som exponerar sitt API globalt via window-objektet (t.ex. window.State = ...).
  + **JavaScript-filer (UI-Komponenter):** Placeras i js/components/. Dessa ska vara ES6-moduler (export const ComponentName = (function() { ... return {init, render, destroy}; })();). Varje komponent ansvarar för sin egen rendering och logik.
    - Komponenter ska ha en async init(container, routerCallback, params)-metod, en render(params)-metod, och en destroy()-metod.
* **CSS-hantering:**
  + En global css/style.css för grundläggande styling (typsnitt, färgschema, globala klasser, formulärstyling).
  + Varje JavaScript UI-komponent (i js/components/) ansvarar för att dynamiskt ladda sin specifika CSS-fil (t.ex. css/components/sample\_list.css) via Helpers.load\_css() i sin init()-metod.
  + CSS-variabler ska användas flitigt för temahantering (färger, typsnitt, etc.).
* **Rendering:** index.html ska endast innehålla minimala placeholders (t.ex. <div id="app-container"></div>). All rendering av vyer och innehåll sker dynamiskt via JavaScript-komponenter som renderar sitt innehåll i #app-container.
* **URL-hantering (Routing):**
  + Använder URL-hash (#) för att styra vilken vy som visas.
  + main.js hanterar hashchange-event och anropar en funktion (t.ex. render\_view(viewName, params)) som byter ut innehållet i #app-container.
  + Komponenter som behöver navigera anropar en funktion (t.ex. navigate\_and\_set\_hash(viewName, params)) som tillhandahålls av main.js, vilken både uppdaterar hashen och triggar vybyte.

**3. Allmänna Krav och Principer**

* **Datauthållighet (session):** Appen ska klara F5 (sidomladdning) och webbläsarens bakåt-/framåtknappar utan att förlora pågående granskningsdata. Detta hanteras via sessionStorage för current\_audit-objektet.
* **Kodläsbarhet:** Koden ska vara lättläst och välkommenterad.
  + Använd snake\_case för egna variabler och funktioner i JavaScript.
  + Använd camelCase för DOM API:er och nycklar i JSON-data (undantag kan göras för JSON från externa källor).
  + Event delegation ska användas där det är lämpligt för att hantera händelser på dynamiskt genererat innehåll.
* **Autentisering:** Ingen autentisering i denna version.
* **Responsiv Design:** Primärt designad för 1080p-skärmbredd, men ska vara funktionell och användbar även i smalare vyer (ner till ca 320px bredd för kritiska vyer).
* **Designestetik:** Modern, professionell och färgglad (t.ex. Teal/Coral-tema som diskuterat). Använd rundade hörn (var(--border-radius)) och accentfärger konsekvent. "Plattor" (.content-plate) för att gruppera relaterat innehåll i vyer.
* **Typsnitt:** Endast sans-serif-typsnitt. Roboto används initialt.
* **Knappdesign:** Alla knappar ska ha:
  + Distinkta färger baserat på funktion (primär, sekundär, default, accent, fara, varning, info).
  + En SVG-ikon (från Helpers.get\_icon\_svg()) till vänster om texten.
  + Ikonen ska ha aria-hidden="true" (hanteras av Helpers.get\_icon\_svg()).
  + Texten ska vara beskrivande. Inga title-attribut om texten är tillräcklig.
* **Checkboxar:** Ska initialt använda webbläsarens default-rendering för att säkerställa funktionalitet. (Custom styling kan återbesökas senare om tid finns).
* **Globala UI-kontroller:** Språkväljare och Tema-toggle ska finnas globalt tillgängliga (t.ex. i ett sidhuvud eller fast position).
  + Tema-toggle ska visa ikon och text för det tema som *kommer att aktiveras*.

**4. Kärnfunktionalitet och Arbetsprocess (Huvudpunkter från tidigare, med förtydliganden)**

* **4.1 Start och Initialisering (UploadViewComponent):**
  + Knappar: "Ladda upp pågående granskning", "Starta ny granskning".
  + Uppladdning av Regelfil (JSON) vid "Starta ny granskning".
  + **Validering av Regelfil (ValidationLogic.js):**
    - Giltig JSON.
    - Schema: Toppnivånycklar metadata, requirements. Inuti metadata: title (obligatorisk sträng), pageTypes (array av strängar, obligatorisk), contentTypes (array av objekt {id, text}, obligatorisk).
    - requirements: Objekt där varje nyckel är en UUID och värdet är ett kravobjekt.
    - Kravobjekt: id (unik sträng), title (sträng), expectedObservation (sträng), checks (array, kan vara tom), contentType (array av strängar, kan vara tom).
    - Varje check: id (sträng), condition (sträng), passCriteria (array av objekt {id, requirement}). logic ("OR" / "AND") är optionell.
  + Vid lyckad validering: Spara i current\_audit, navigera till Metadata-vyn.
  + Felmeddelanden visas i det globala meddelandefältet (hanteras av NotificationComponent).
* **4.2 Inmatning av Metadata (MetadataViewComponent):**
  + Formulär: Ärendenummer, Aktör, Länk (auto-https), Ansvarig, Intern kommentar (alla frivilliga).
  + Texten "Ange metadata för granskningen. Alla fält är frivilliga." visas direkt under H1-rubriken.
  + Inga placeholders i inputfälten.
  + Knapp: "Fortsätt till stickprov".
  + Fält låsta om auditStatus !== 'not\_started'.
* **4.3 Hantering av Stickprov (SampleManagementViewComponent, AddSampleFormComponent, SampleListComponent):**
  + **SampleManagementView:**
    - Knapp "Lägg till nytt stickprov" / "Visa befintliga stickprov" för att växla vy.
    - Knapp "Starta granskning" visas endast om samples.length > 0 och auditStatus === 'not\_started'.
    - Visar initialt formuläret om inga stickprov finns, annars listan.
  + **AddSampleForm:**
    - Typ av sida (dropdown från ruleFileContent.metadata.pageTypes).
    - Beskrivning (auto-ifylls från Typ av sida om beskrivning är tom eller matchar föregående sidtyp, annars behålls användarens text).
    - URL (auto-https).
    - Innehållstyper (checkboxar från ruleFileContent.metadata.contentTypes, minst en vald).
    - Knappar: "Spara stickprovet" (eller "Spara ändringar" vid redigering) och "Visa befintliga stickprov" på samma rad.
  + **SampleList:**
    - Visar: Beskrivning (H3), Typ av sida, URL (klickbar), Hanterade/Totala krav (placeholder för nu), Innehållstyper (lista).
    - Knapprad (om auditStatus === 'not\_started'): "Redigera", "Radera".
    - Knapprad (alltid, om relevanta krav finns): "Se alla krav" (leder till RequirementListComponent).
    - Knapprad (alltid, om URL finns): "Besök sidan".
    - Knapprad (om auditStatus === 'in\_progress' och relevanta krav finns): "Granska" (leder till första oreviderade krav i RequirementAuditComponent - TODO).
  + **Flöde:** Efter att ett stickprov sparats i AddSampleFormComponent stannar användaren i SampleManagementViewComponent, formuläret döljs, listan visas med det nya/uppdaterade stickprovet.
* **4.4 Starta Granskning (från SampleManagementViewComponent):**
  + Vid klick på "Starta granskning":
    - auditMetadata blir effektivt låst (UI i MetadataView visar statisk text).
    - startTime sätts. auditStatus blir in\_progress. Spara till State.
    - Navigera till AuditOverviewComponent.
    - Redigeringsmöjligheter för stickprov och "Lägg till nytt stickprov" inaktiveras.
* **4.5 Genomföra Granskning:**
  + **RequirementListComponent:**
    - Visar relevant info om valt stickprov (Namn H1, Typ av sida).
    - Listar relevanta krav, grupperade efter mainCategory.text (H2) och subCategory.text (H3). Varje kravtitel som H4.
    - Varje kravs titel är **inte** klickbar här. Standardreferens visas som länk.
    - Knappar (uppe och nere): "Tillbaka till alla stickprov" (om auditStatus === 'not\_started', leder till SampleManagementView) eller "Tillbaka till granskningsöversikten" (om auditStatus === 'in\_progress' eller 'locked', leder till AuditOverviewComponent).
    - Knapp "Godkänn alla återstående..." (om auditStatus !== 'locked').
  + **RequirementAuditComponent (TODO):** (Detaljer från spec 5.6)
    - Visar all kravinformation.
    - Renderar checks och passCriteria med "Godkänt"/"Underkänt"-knappar.
    - Inmatningsfält för observationer/kommentarer.
    - Navigeringsknappar.
    - Automatisk sparning till sessionStorage.
* **4.6 Avsluta och Låsa Granskning (från AuditOverviewComponent):**
  + Knapp "Lås granskning" (om auditStatus === 'in\_progress').
  + Sätter endTime, auditStatus till locked. Inaktiverar redigering.
  + Knapp "Lås upp granskning" (om auditStatus === 'locked').
* **4.7 Spara/Ladda Pågående Granskning till/från Fil:**
  + **Spara (från AuditOverviewComponent):** Knapp "Spara granskning till fil". Serialiserar current\_audit. Filnamn: granskning\_[regelfilstitel]\_[datum\_YYYY-MM-DD-HH-MM-SS].json.
  + **Ladda (från UploadViewComponent):** Knapp "Ladda upp pågående granskning". Validerar fil (JSON, korrekt struktur). Återställer current\_audit. Navigerar till AuditOverviewComponent.
* **4.8 Export av Resultat (från AuditOverviewComponent, endast vid låst granskning):**
  + CSV och Excel (XLSX via SheetJS). Detaljer enligt tidigare spec.

**5. Vystruktur och Logik (Översikt)**

* **5.1 Startvy (UploadViewComponent):** H1, intro, knappar. Globalt meddelandefält för fel.
* **5.2 Metadata-vy (MetadataViewComponent):** Formulär/statisk text. Knapp "Fortsätt...". Globalt meddelandefält + text under H1. Allt på en "platta".
* **5.3 Stickprovs-vy (SampleManagementViewComponent):** Knapp "Lägg till/Visa...". Innehåller AddSampleFormComponent och SampleListComponent. Knapp "Starta granskning". Allt på en "platta".
* **5.4 Översiktsvy (AuditOverviewComponent):**
  + Sektion 1: Granskningsinfo (metadata, regelfil, status, tider).
  + Sektion 2: Stickprovslista (via SampleListComponent). Knappar anpassade för översiktskontext.
  + Sektion 3: Åtgärdsknapprad ("Lås/Lås upp", "Spara fil", "Exportera").
* **5.5 Kravlistvy (RequirementListComponent):** Stickprovsnamn (H1), info. Grupperad lista med krav (H2, H3, H4). "Tillbaka"-knappar.
* **5.6 Kravgransknings-vy (RequirementAuditComponent):** Detaljerad kravinfo, checks, inmatningsfält, nav-knappar.

**6. Datahantering och Lagring (State.js, sessionStorage)**

* current\_audit-objektets struktur: Som specificerat, inklusive saveFileVersion: "1.2".
  + samples: Array av { id, pageType, description, url, selectedContentTypes, requirementResults: { [req.id]: resultObject } }.
  + resultObject: { status ("passed", "failed", "not\_audited"), actualObservation, commentToAuditor, commentToActor, lastStatusUpdate, checkResults: { [check.id]: { status, passCriteria: { [pc.id]: "passed/failed/not\_audited" } } } }.
* Automatisk sparning i RequirementAuditComponent.
* Tidsstämplar ISO 8601 (UTC).
* localStorage för tema.

**7. Tillgänglighet (WCAG 2.2 AA)**

* Som specificerat. Inga disabled komponenter (använd aria-disabled och visuell inaktivering istället, eller dölj/ta bort).
* Korrekt label-hantering. Full tangentbordsnavigering. Färgkontrast.
* Dark Mode (OS + manuell toggle).

**8. Kodstruktur och Moduler (Översikt)**

* index.html, css/style.css.
* js/: main.js, state.js, audit\_logic.js, export\_logic.js, validation\_logic.js, utils/helpers.js.
* js/i18n/: sv-SE.json, en-GB.json. translation\_logic.js (laddar JSON).
* js/components/: UI-komponenter som ES6-moduler, egen CSS.
* Globalt meddelandefält hanteras av NotificationComponent.js och infogas i aktuella vyns DOM.

**9. Internationalisering (i18n)**

* All UI-text via Translation.t("key") från laddade JSON-filer.
* Initialt svenska, förberett för engelska. Språkväljare i UI.
* Dynamisk uppdatering av UI vid språkbyte.

**10. JSON Regelfil (Användning och Validering)**

* Validering enligt punkt 4.1.1. Struktur med metadata (innehållande pageTypes, contentTypes) och requirements (objekt).

**11. Potentiella framtida utökningar (Ej i denna version)**

* Mer avancerad sortering/filtrering i listor.
* Drag-and-drop för att ordna stickprov.
* Realtidssamarbete (kräver backend).
* Mer avancerad rapportgenerering.
* Integration med externa system.
* Utökad schemavalidering för regelfil.

**Frågor till dig/Saker att fundera på:**

* **Checkbox-styling:** Är du nöjd med default-checkboxar för nu, eller vill du att vi gör ett nytt försök med custom styling senare?
* **Knappen "Granska" i SampleListComponent (när den visas i AuditOverviewComponent):** Vad exakt ska den göra? Ska den leda till RequirementListComponent (som "Se alla krav") eller direkt till RequirementAuditComponent för det första oreviderade kravet för det stickprovet? Det senare är mer komplext att implementera initialt.
* **Sortering av krav:** Är sorteringen mainCategory.text, subCategory.text, title tillräcklig, eller behövs mer avancerade sorteringsalternativ? (Detta kan vi ta senare).

Jag har försökt vara så detaljerad som möjligt. Stämmer detta överens med hur du ser på applikationen nu? Finns det något uppenbart jag har missat eller missförstått från våra senaste diskussioner?