Polish — Sprint 2 Test Plan

AI Résumé Builder with Inline Chatting

Date: November 5, 2025

1. Introduction

This test plan documents the approach for validating Sprint 2 of Polish, an AI résumé builder that enables line-by-line inline chat assistance. It covers objectives, scope, risks, test design/automation, environments, milestones, and detailed test cases for the Sprint 2 backlog.

1.1 Objectives

Validate that Sprint 2 increments are functional, reliable, secure, and ready for demo: (a) Azure backend services are integrated and stable, (b) user authentication (sign-up, sign-in, sign-out, and session) is implemented, and (c) an Editorial page enables creation, editing, and saving of résumé content with inline chat per line.

1.2 Team Members

Resource Name	Role
Mohammed	Developer
Arnav	Project Manager / Tester
Walid	Tester
Matt	Developer; Project Manager / Tester

2. Scope

This plan focuses on Sprint 2 deliverables and regression touchpoints from Sprint 1 that affect the new work. Must-have requirements for Sprint 2 include Azure backend integration, Authentication flows, and the Editorial page with inline chat. Non-functional testing includes basic security, performance smoke, and accessibility checks on changed surfaces.

At the end of Sprint 2, a user must be able to:

- Create an account, sign in, and sign out using the Authentication page; maintain a valid session.
- Access the Editorial page to create/edit a résumé and persist changes to the Azure backend.
- Invoke inline chat assistance for any résumé line and view responses without breaking editing.
- Reload the app and see previously saved content restored from the backend.

Additional testing in Sprint 2:

- Security: auth guard, unauthorized access, basic input validation.
- Performance: initial load and chat-response latency smoke checks.
- Accessibility: keyboard navigation and focus management on auth/editorial forms.
- Cross-browser: Chrome, Edge, and Safari latest on desktop; mobile viewport smoke.

3. Assumptions / Risks

3.1 Assumptions

- Azure resources (API, data store, identity) are provisioned and reachable from test networks.
- Feature flags or environment variables allow pointing the web app at QA and Staging backends.
- Inline chat uses a stable model/endpoint with configured quotas.
- Test accounts and seed data are available.

3.2 Risks

#	Risk	Impact	Trigger	Mitigation Plan
1	Scope creep as functionality becomes clearer	High	Backlog growth mid-sprint	Enforce definition-of-done; freeze scope after sprint planning; defer to Sprint 3.
2	External dependency on Azure services	Medium	Regional outage / rate limiting	Introduce retries/backoff; feature-level circuit breakers; have local mock server for testing.
3	Auth integration issues (token expiry, CORS)	Medium	401/403 spikes in QA	Add token refresh and CORS preflight tests in CI; monitor network errors; fallback to logout.
4	Chat cost/latency variability	Medium	High p95 response time or quota exceeded	Cache identical prompts per line; show non-blocking spinners; limit retries; daily quota watch.
5	Accessibility regressions on new pages	Low	Keyboard traps or missing labels	Add axe-core scan in CI; manual keyboard

	walkthroughs per build; fix before merge.

4. Test Approach

We follow 3-week sprints. Testing is continuous: story-level acceptance criteria are turned into black-box test cases; unit and integration tests are added with code. Each PR requires automated checks and reviewer sign-off. We timebox exploratory testing late in the sprint and run a regression suite before the Sprint 2 demo.

4.1 Sprint 2 Approach

1. Requirements → write acceptance criteria and happy/edge cases per user story.
2. Unit tests → auth guards, API clients, editor state reducers, chat adapters.
3. Integration/API tests → Azure endpoints (CRUD résumé, chat relay), auth flows.
4. E2E tests → sign up/in/out, editorial edit/save/restore, inline chat per line, session persistence.
5. Security/access control → unauthorized access attempts to editor and APIs.
6. Accessibility → keyboard-only, form labels, focus order, contrast checks on changed pages.
7. Cross-browser/viewport sanity on desktop and mobile widths.
8. Exploratory testing targeting auth state transitions and editor-chat interplay.
9. Regression sweep on Sprint 1 surfaces impacted by backend/auth changes.

5. Test Automation

Automation layers: (1) Unit tests (framework per stack; e.g., Jest/React Testing Library or equivalent), (2) API tests (Postman/Newman or supertest), (3) E2E flows (Playwright). CI runs on each PR and on main nightly. Accessibility checks via axe-core/Playwright. Basic performance smoke measures via Playwright timings and API p95 logs.

6. Test Cases (Black Box)

6.1 Authentication

Test Case ID	Description	Requirements Trace	Directions	Expected Output
TC-AUTH-001	Sign up with valid credentials	AUTH-001	Open app → Go to Auth page → Fill valid email/password → Create account.	User account created; redirected to Editorial page; session cookie/token set.
TC-AUTH-002	Sign in existing	AUTH-002	Open app →	Redirect to

	user		Auth page → Enter valid credentials → Sign in.	Editorial; session active; user name visible.
TC-AUTH-003	Reject invalid password policy	AUTH-003	Attempt sign up with weak password.	Inline validation shows policy error; account not created.
TC-AUTH-004	Sign out clears session	AUTH-004	From Editorial, choose Sign out.	Returned to Auth page; accessing Editorial requires re-auth.
TC-AUTH-005	Session expiry/refresh	AUTH-005	After token expiry simulate API call.	Silent refresh works or user is prompted to sign in again.

6.2 Editorial Page

Test Case ID	Description	Requirements Trace	Directions	Expected Output
TC-EDIT-001	Create new résumé	EDIT-001	From Editorial, click New → enter headline and first section.	Draft created with untitled name; auto-save enabled.
TC-EDIT-002	Edit and save content	EDIT-002	Modify a line; pause 1–2s.	Auto-save to Azure backend; save indicator confirms.
TC-EDIT-003	Reload restores content	EDIT-003	Hard refresh browser.	All content restored from backend; caret position reasonable.
TC-EDIT-004	Undo/redo editing	EDIT-004	Make changes then undo/redo.	Edits revert/restore without data loss.
TC-EDIT-005	Unauthorized access blocked	SEC-002	Open /editorial route while signed out.	Redirected to Auth; 401/403 on protected APIs.

6.3 Inline Chat per Line

Test Case ID	Description	Requirements	Directions	Expected

		Trace		Output
TC-CHAT-001	Request suggestions for a line	CHAT-001	Select a résumé line → Click 'Ask AI'.	Chat panel opens with suggestions contextual to selected line.
TC-CHAT-002	Apply suggestion	CHAT-002	Click 'Apply'.	Line text updates; change is auto-saved.
TC-CHAT-003	Rate/undo suggestion	CHAT-003	Rate suggestion or undo.	Feedback recorded; content reverts if undone.
TC-CHAT-004	Error/timeout handling	CHAT-004	Simulate API error or slow response.	Non-blocking error message; retry option; editor remains usable.
TC-CHAT-005	Quota exceeded scenario	CHAT-005	Simulate quota cap.	Graceful degradation message; prevents repeated calls within cooldown.

6.4 Azure Backend Integration

Test Case ID	Description	Requirements Trace	Directions	Expected Output
TC-AZ-001	Create/read/update résumé via API	AZ-CRUD-001	Use app to create and edit; observe network.	200 responses; persisted data consistent with UI.
TC-AZ-002	Handle network loss	AZ-RES-002	Toggle offline mid-edit.	Changes queued; user warned; sync resumes on reconnect.
TC-AZ-003	Authorization on APIs	SEC-API-003	Call APIs without token.	401/403; no data leaked.
TC-AZ-004	Data model validation	AZ-VAL-004	Post malformed payload via API test client.	400 with clear validation errors; no server crash.
TC-AZ-005	Basic performance smoke	PERF-005	Measure p95 for save and chat relay.	p95 save < 800ms; chat relay < 2.5s

(targets).

7. Test Cases (White Box)

Test Case ID	Description	Directions/Goals	Expected Output
TC-WB-001	Auth guard unit tests	Simulate routes with/without tokens; token refresh edge cases.	Correct redirects; refresh paths covered.
TC-WB-002	Editor reducers/selectors	Fuzz edits, undo/redo, autosave triggers.	Deterministic state; no orphan timers.
TC-WB-003	Chat adapter	Mock chat service; test retries/backoff and cancellation.	Retries bounded; cancellation respected.
TC-WB-004	API client	Mock Azure endpoints; validate error mapping.	Consistent error objects; no unhandled rejections.
TC-WB-005	Accessibility unit checks	Snapshot tests for aria-* attributes and labels.	All interactive controls labeled; focus traps avoided.

8. Test Environment

Environments: Dev (feature branches), QA (main branch), and Staging (pre-demo). Azure resource group per env. Browsers: Chrome, Edge, Safari (latest major). Desktop viewport 1440×900; mobile viewport 390×844 for smoke. Test data: seeded demo users and sample résumés. Network controls to simulate latency/offline.

9. Milestones / Deliverables

9.1 Test Schedule (Sprint 2: Oct 16, 2025 – Nov 5, 2025)

Task Name	Start	Finish	Effort	Comments
Test Planning	Oct 16	Oct 17	1d	Identify AC and risks.
Review Requirements	Oct 16	Oct 18	2d	Trace stories AUTH/EDIT/CHAT/AZ-*.
Create initial test estimates	Oct 18	Oct 18	0.5d	Coverage & automation split.
Learn new test resources	Oct 18	Oct 20	2d	Playwright setup; API fixtures.
First deploy to QA	Oct 21	Oct 21	0.5d	Smoke & env checks.
Functional	Oct 21	Oct 31	7d	Black-box story tests.

testing – Sprint 2				
Iteration 2 deploy to QA	Oct 28	Oct 28	0.5d	Mid-sprint fixes.
System testing	Nov 1	Nov 2	2d	End-to-end flows across features.
Regression testing	Nov 2	Nov 3	1d	Sprint 1 surfaces touched by changes.
Usability Testing	Nov 3	Nov 3	0.5d	Editorial & chat UX walk-through.
Resolution of final defects & final build testing	Nov 3	Nov 4	1d	Close P1/P2s.
Deploy to Staging	Nov 4	Nov 4	0.5d	Go/no-go checklist.
Performance testing (smoke)	Nov 4	Nov 4	0.5d	p95 save/chat targets.
Sprint 2 Demo / Release	Nov 5	Nov 5	0.5d	Stakeholder demo.

9.2 Deliverables

- Updated test plan and traceability to Sprint 2 stories (this document).
- Automated test suites: unit, API, and E2E (CI artifacts).
- Test summary report with defect list and pass/fail status.
- Accessibility and performance smoke results.

— End of Sprint 2 Test Plan —