AUTOMATION WORKFLOW

NOLA Puzzle Automation

Daily puzzle delivery automation

A browser automation that retrieves Wonderword and Jumble from a subscriber's digital newspaper, validates the content, prepares a printable PDF and delivers it automatically each weekday.

ROLE

Workflow architecture, browser automation, testing, client handoff

STACK

Python, Playwright, persistent browser session, Windows Task Scheduler

STATUS

Client MVP

E-Edition → Locate Puzzles → Validate → Crop → Build PDF → Deliver

The Problem

A friend had a simple but repetitive weekday routine. Each morning, he opened his NOLA.com digital newspaper subscription, navigated to the puzzles, located Wonderword and Jumble, clipped both and prepared them for printing.

The process took about 10 minutes a day. None of the individual steps were difficult, but repeating the same sequence every weekday made it a natural candidate for automation.

The challenge was that the puzzles lived inside an authenticated digital newspaper experience rather than on predictable public webpages. Any solution would need to work through the subscriber's legitimate session, navigate the publication and reproduce the useful end result without requiring him to repeat the manual process.

> **DESIGN GOAL**
>
> Build a simple workflow that retrieves the correct puzzles, prepares a printable PDF and delivers it automatically with minimal ongoing effort.

CASE STUDY

The Approach

The first step was to map the manual routine exactly as the user performed it rather than redesigning the process around the automation.

The intended workflow was straightforward: open NOLA.com, enter the E-Edition, log in, select the current edition, navigate to Comics/Puzzles, locate Wonderword and Jumble, clip both puzzles, combine them into a PDF and send the finished file by email.

Browser automation with Python and Playwright was a natural fit because the task depended on interacting with the same website interface the subscriber already used.

Authentication added an early constraint. Instead of trying to automate credentials or bypass the subscription flow, I used a persistent browser profile so the automation could reuse the subscriber's authenticated session.

That kept the solution aligned with the existing user experience while removing the repetitive navigation and preparation steps.

Key Design Decision: Automate the user's existing path through the newspaper rather than build a separate way around it.

CASE STUDY

The First Working Solution

Once the browser could reproduce the E-Edition access and login sequence, the manual routine could be automated from beginning to end.

Using the persistent browser session, the automation opened NOLA.com, entered the E-Edition, navigated to Comics/Puzzles and reached the pages containing Wonderword and Jumble. It then extracted the puzzle areas, prepared them for printing and combined them into a single PDF.

Email delivery completed the workflow, turning a sequence of browser interactions into one automated process.

At this stage, the system appeared ready. The expected weekday pattern was consistent, the correct puzzles were being captured and the finished PDF matched the result the user had previously created by hand.

Core Workflow

NAVIGATE

Enter the E-Edition and reach the current Comics/Puzzles section.

EXTRACT

Locate and capture Wonderword and Jumble for printing.

DELIVER

Combine both puzzles into one PDF and send the finished file by email.

The remaining question was whether the same assumptions would continue to hold across different editions.

CASE STUDY

The Real-World Curveball

The first successful runs exposed a dangerous assumption: a workflow that worked for one newspaper edition was not necessarily reliable across others.

As I tested additional days, the newspaper layout changed. Page numbers varied and the position of the puzzles could shift. Logic that depended too heavily on a known page or fixed location could work perfectly one day and capture the wrong content the next.

Rather than continue patching individual editions, I went back into the reader to understand what remained consistent underneath the changing presentation. That investigation shifted the solution away from brittle page assumptions and toward the content and metadata associated with each puzzle.

Historical testing then became part of the development process. Wonderword and Jumble were tested across multiple prior editions to determine whether the new approach held when the visible newspaper layout changed.

Testing also exposed a boundary: the Sunday edition followed a substantially different structure. Because the immediate client workflow only required weekdays, I treated Sunday as a known exception rather than expanding the project to solve a problem the client did not need solved.

Key Design Decision: When the presentation proved variable, stop designing around where the puzzles appeared and start designing around how they could be reliably identified.

How the Finished System Works

The finished workflow combines browser automation, content discovery, validation and document delivery into a single process.

01 · Start
Launch the automation on schedule or manually.

02 · Authenticate
Open NOLA.com through the subscriber's persistent browser session.

03 · Navigate
Open the current E-Edition and reach the relevant puzzle content.

04 · Identify
Locate Wonderword and Jumble using the content information discovered during development rather than relying on fixed page numbers.

05 · Validate & Extract
Confirm the expected puzzle information is present, then apply the corresponding crop to each source page.

06 · Build
Format both puzzles into a clean two-page printable PDF.

07 · Deliver
Email the completed PDF to the user.

Scheduled Run → Authenticated Session → E-Edition → Puzzle Discovery → Validation & Extraction → PDF → Email

Fail-Safe Design & Testing

Once the workflow could identify puzzles dynamically, the next priority was making sure it failed safely when its assumptions were no longer valid.

A successful browser run was not enough. The automation needed evidence that it had found the expected puzzle content before creating and emailing the final PDF. Validation checks were added around the discovered puzzle information and crop data so an unexpected result could stop the workflow instead of silently producing the wrong output.

Reliability was tested against multiple historical editions rather than a single successful day. Wonderword and Jumble were checked across September 5–9, including editions with changing page locations. The same approach continued to work across the standard layouts, while Sunday testing exposed the different structure that was intentionally left outside the production scope.

After the workflow was assembled end to end, it was tested again with the current edition and through the same scheduled execution path the client would use.

FAIL-SAFE PRINCIPLE

If the system cannot validate what it found, stop rather than send a plausible-looking but incorrect result.

Production Output

The completed workflow produces a print-ready two-page PDF containing the correct Wonderword and Jumble for the current edition.

The output is the final validation point: the automation must successfully identify, extract and prepare both puzzles before the file is delivered.

A successful run turns the original multi-step newspaper routine into one finished file, ready to print.



WONDERWORD

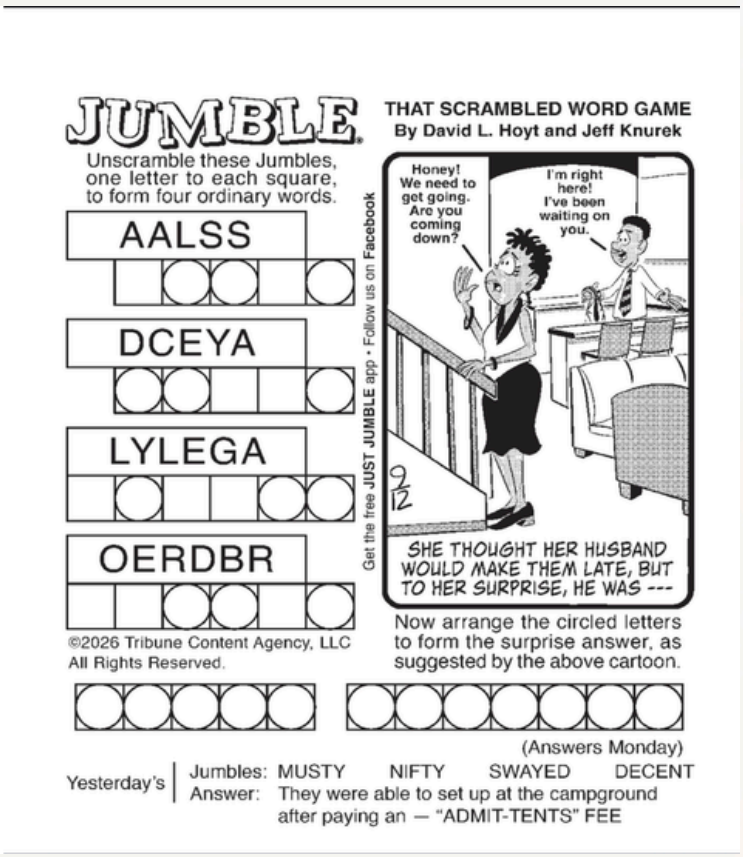
TOMMY HUNTER (1937-2026) Solution: 9 letters

F	A	M	I	L	Y	R	T	N	U	O	C	L	A	P
F	N	R	E	T	S	E	W	A	R	A	B	R	A	B
E	O	E	S	E	R	U	S	A	E	R	T	E	Y	C
J	C	G	O	O	H	O	E	D	O	W	N	(S)	R	U
L	I	N	I	G	N	C	S	E	C	A	(E)	P	A	P
A	P	I	R	E	E	G	A	N	M	(M)	V	E	S	O
N	E	S	A	N	T	D	S	N	(A)	L	O	C	R	F
O	R	O	T	N	E	M	I	(J)	A	L	L	T	E	D
I	F	G	N	E	S	L	W	T	U	D	P	E	V	I
T	O	E	O	T	L	O	Y	L	H	D	I	D	I	S
I	R	R	A	E	R	L	M	A	O	U	D	A	N	G
D	M	G	V	L	E	P	S	O	G	N	M	S	N	R
A	E	A	D	R	N	L	E	G	E	N	D	B	A	A
R	R	E	T	S	A	C	D	A	O	R	B	O	L	C
T	S	I	R	A	T	I	U	G	E	N	U	I	N	E

© 2026 Andrews McMeel Syndication www.wonderword.com 9/12

Anniversary, Barbara, Broadcaster, Canadian, Clap, Country, Cup of Disgrace, Edith, Family, Genuine, Gospel, Greg, Guitarist, Hoedown, Humble, Icon, James, Jeff, Judds, Legend, London, Love, Mentor, Ontario, Peace, Performer, Plans, Respected, Singer, Songs, Stage, Traditional, Travellin' Man, Treasures, Tyler, Western, World

Yesterday's Answer: Hancock



JUMBLE®

THAT SCRAMBLED WORD GAME
By David L. Hoyt and Jeff Knurek

Unscramble these Jumbles, one letter to each square, to form four ordinary words.

AALSS

DCEYA

LYLEGA

OERDBR

©2026 Tribune Content Agency, LLC All Rights Reserved.

Get the free JUST JUMBLE app • Follow us on Facebook

Now arrange the circled letters to form the surprise answer, as suggested by the above cartoon.

(Answers Monday)

Yesterday's Jumbles: MUSTY NIFTY SWAYED DECENT
Answer: They were able to set up at the campground after paying an — "ADMIT-TENTS" FEE

Scheduling & Human Control

Once the core workflow was reliable, the final automation decision was how much control to leave with the user.

The default experience is hands-off. Windows Task Scheduler launches the workflow at 4:00 AM Monday through Friday so the puzzles can arrive automatically before the user needs them.

I also preserved a manual path. The same production workflow can be launched by double-clicking a batch file, allowing the user to request the puzzles on demand if the computer was off overnight, the scheduled run was missed or he simply wants them later.

The persistent browser session keeps routine runs simple, but authentication remains an external dependency. If NOLA.com expires the session, the user may occasionally need to sign in again before automation can resume.

The goal was not maximum automation. It was a workflow that remained convenient when normal routines changed.

OPERATING MODEL

Scheduled by default · Manual when useful · Reauthenticate when required

Production & Handoff

A working automation was only useful if the client could operate it without depending on my development environment.

I separated the finished production files from the scripts, test outputs and experiments created during development. The client received a clean handoff package containing the production automation, a simple launcher, setup documentation and a starter prompt for guided installation.

Configuration was designed to remain client-owned. The user supplies his own email credentials and authenticates his own NOLA.com subscription locally rather than receiving credentials or browser-session data from my machine.

The handoff also documents the most likely maintenance issue: an expired NOLA.com session. Troubleshooting begins by checking authentication before assuming the automation itself has failed.

Handoff Principle: Deliver the smallest useful production package, document the known dependencies and keep development artifacts out of the client's working environment.

The Result

The finished system replaced a repetitive weekday routine with a client-ready workflow that can run automatically or on demand.

The automation now handles the full path from authenticated newspaper access through puzzle discovery, validation, extraction, PDF creation and email delivery.

The finished system:

- Retrieves Wonderword and Jumble from the current digital edition
- Uses a persistent browser session for authenticated access
- Identifies puzzle content without depending on fixed page numbers
- Validates expected information before creating the output
- Produces a print-ready two-page PDF
- Delivers the completed file by email
- Runs automatically at 4:00 AM Monday through Friday
- Preserves a manual launcher for on-demand use

Before handoff, the workflow was tested through its production execution path and the client received the resulting PDF. He confirmed that the finished puzzles looked correct.

Final Workflow:

Schedule or Manual Launch → Retrieve → Validate → Extract → Build → Deliver

What began as a small convenience project became a practical example of designing automation around the variability of a real external system.

What I Learned

The biggest lesson from this project was that a successful automation run does not necessarily mean the underlying workflow is reliable.

The manual process took only about 10 minutes, which initially made the project appear straightforward. Testing across different newspaper editions exposed the real challenge: changing layouts meant that assumptions that worked one day could fail the next.

The project reinforced several principles I want to carry into future workflow automation work:

- **Test the assumption, not just the automation:** A successful run proves that a workflow can work. Testing across different conditions reveals whether the assumptions underneath it are actually reliable.
- **Simple tasks can hide variable systems:** The visible task may be repetitive and straightforward even when the system behind it changes from one run to another.
- **Design for the cost of being wrong:** Sending the wrong puzzle was worse than sending nothing, making validation and a safe stop more important than maximizing successful runs.
- **Scope around the real need:** Sunday introduced a substantially different layout, but the client needed weekday automation. Treating it as a known exception kept the project focused.
- **Estimate uncertainty, not just the successful outcome:** External dependencies, testing and unexpected behavior can require more effort than the apparent number of workflow steps suggests.

The project reinforced a broader lesson: reliability often comes from discovering which early assumptions should no longer be trusted.

Next Version

The current system solves the client's actual weekday need, so the next version would focus on resilience and portability rather than adding features for their own sake.

Stronger Layout Resilience
Expand validation and discovery logic if future newspaper changes introduce new layouts or puzzle structures.

Sunday Support
Investigate the substantially different Sunday edition only if weekend automation becomes useful to the client or a future user.

Easier Installation
Reduce the number of environment-specific setup steps so the production package can be transferred to another Windows computer with less configuration.

Deployment Flexibility
Evaluate cloud execution only when a use case genuinely requires an always-available machine. The current combination of scheduled and manual execution avoids adding infrastructure the client does not presently need.

Broader Reuse
If the workflow is adapted for another user, treat their subscription, puzzle preferences and delivery requirements as configuration rather than assuming the original client's routine applies.

The next step is not to automate more. It is to make the existing workflow easier to transfer, maintain and adapt when there is a real reason to do so.