PROJECT TECHNICAL REPORT

temp-mail-agent

AI-Powered Cross-Platform Account Registration Automation Agent

Version: 2.0.0 | **Date:** April 2026 | **Platform:** Windows / macOS / Linux

1. Project Overview

temp-mail-agent is a Node.js automation agent that combines a local large language model (Llama 3.2 via Ollama), a headless Chrome browser (Puppeteer), and temporary email generation to automatically complete web registration forms. The agent interacts with users through voice input and text-to-speech feedback, simulates human-like mouse movements and keystrokes to avoid bot detection, and falls back to heuristic logic when the AI model is unavailable.

The v2.0 release resolves critical bugs present in v1 and introduces full cross-platform support for Windows, macOS, and Linux with no code changes required between platforms.

2. System Architecture

The project is structured around a single main orchestration file (agent.js) with a supporting Python script for audio capture. The system has five logical layers:

Layer	Responsibility
Platform Layer	OS detection, Chrome path resolution, TTS via native OS commands
I/O Layer	Voice input (Python + SpeechRecognition), text fallback via readline
Browser Layer	Puppeteer + Stealth plugin — launches Chrome, spoofs navigator APIs
AI Decision Layer	Llama 3.2 (Ollama) reads the page DOM and returns JSON actions
Fallback Layer	Heuristic keyword matching when Ollama is offline or unavailable

3. File-by-File Breakdown

3.1 agent.js (Main Brain — ~430 lines)

The central orchestration file. Responsibilities are divided across clearly scoped functions:

Function	Purpose
getChromePath()	Detects OS and returns the correct Chrome binary path
speak(text)	Cross-platform TTS: say (macOS), PowerShell (Windows), espeak (Linux)
listenVoice()	Spawns listen.py and parses the transcribed text
ask(query)	Speaks the question, waits for voice, falls back to typed input
generatePassword()	Produces a cryptographically random 16-char password
humanMove(page, sel)	Moves mouse to element with random offset + smooth 25-step path
humanType(page, sel, txt)	Fires React-compatible native input events + char-by-char keyboard
extractInteractiveElements	Scans visible inputs, selects, buttons, and assigns stable IDs
extractPageErrors()	Finds visible error text (taken, invalid, already in use, etc.)
askLlama()	Builds structured prompt, calls Ollama, returns parsed JSON actions
heuristicFallback()	Rule-based field matching when Ollama is offline
getTempEmail()	Tries 1secmail API first, falls back to Temp-Mail.org browser scrape
collectProfile()	Prompts user for identity info, saves/loads from user_profiles.json
main()	Orchestrates the full pipeline: profile → email → navigate → AI loop

3.2 listen.py (Voice Capture)

A self-contained Python script that records 5 seconds of microphone audio using sounddevice, converts it to a SpeechRecognition AudioData object, and submits it to Google's Speech Recognition API. Output is printed as TEXT: <transcription> which agent.js parses via regex.

Gracefully handles three failure modes:

- Missing Python packages — prints an error code instead of crashing
- Microphone unavailable — prints an audio error code
- Unclear speech — prints TEXT: COULD_NOT_UNDERSTAND

3.3 package.json

Dependency	Version
puppeteer	^24.4.0
puppeteer-extra	^3.3.6
puppeteer-extra-plugin-stealth	^2.11.2

Dependency	Version
ollama	^0.6.3
node-fetch	^3.3.2

The project uses ES Modules ("type": "module" in package.json), requiring Node.js v18 or newer. All imports use the import keyword; dynamic imports are used for util to maintain compatibility.

3.4 Data Files

File	Contents
user_profiles.json	Saved identity profile (name, username, DOB, country, gender, master password)
accounts_db.json	Auto-created log of successfully registered accounts (email, password, website, timestamp)

IMPORTANT: These files contain personal data and credentials. Never include them when sharing the project. The v2 release ships with a blank user_profiles.json. The accounts_db.json is created at runtime.

4. How the Agent Works (Step-by-Step)

Step	Stage	Description
1	Profile Collection	Agent asks for name, username, DOB, country, gender, password preference via voice/text. Saves to user_profiles.json.
2	Target URL Input	User provides the website to register on.
3	Chrome Launch	Puppeteer opens a real Chrome window with stealth settings applied.
4	Temp Email Generation	Calls 1secmail API to get a throwaway email address. Falls back to scraping Temp-Mail.org.
5	Smart Navigation	Tries /register, /signup, /sign-up, /join directly. Falls back to home page if none work.
6	AI Loop (up to 15 steps)	Each step: extract DOM elements + errors, ask Llama 3.2, execute type/click/select actions.
7	Human Confirmation	After each step, asks user by voice: does the form look correct? Yes = submit. No = retry.
8	Credential Logging	On success, saves email + password + website + timestamp to accounts_db.json.

5. AI Decision Engine

5.1 Primary: Ollama + Llama 3.2

At each loop iteration, the agent extracts all visible interactive elements from the page DOM and sends them to a locally-running Llama 3.2 model (via Ollama) along with the user profile, previous actions taken, and any visible error messages.

The model receives a structured JSON prompt and is instructed to return only a JSON object with three fields: a thought (reasoning), a status (progress / success / failed), and a list of actions (type, click, select) with exact element IDs from the DOM. Temperature is set to 0.1 for deterministic, reliable output.

5.2 Fallback: Heuristic Rules

If Ollama is not running or returns an unparseable response, the heuristicFallback() function takes over. It scans element attributes (name, id, placeholder, ariaLabel) for keywords:

Field Keyword Match	Value Used
email, e-mail	profile.email
user, username	profile.username
first, fname	profile.firstName
last, lname	profile.lastName
pass, pwd	profile.password
dob, birth, date	profile.dob
country (select)	profile.country
gender, sex (select)	Matched against select options

6. Cross-Platform Support

Feature	Implementation
Chrome Path	Auto-detected per OS — Windows Program Files, macOS /Applications, Linux /usr/bin
TTS	macOS: say | Windows: PowerShell System.Speech | Linux: espeak (optional)
Voice Input	Python sounddevice + SpeechRecognition — works identically on all platforms
User Agent	Set to Windows Chrome 124 to blend in universally
Fallback	If TTS or voice fails, agent continues silently via text prompts — never crashes

7. Bugs Fixed in v2.0

Critical Bug — Duplicate const page Declaration (v1)

- In agent.js v1, const page was declared twice in the same function scope
- Node.js throws a SyntaxError at startup, preventing the agent from ever running
- Fix: The second declaration is removed; the default browser tab is reused with const [page] = await browser.pages()

macOS-Only Hardcoded Paths (v1)

- Chrome path was hardcoded to /Applications/Google Chrome.app — crashes on Windows and Linux
- TTS used the macOS say command — not available on other platforms
- Fix: getChromePath() detects OS at runtime; speak() uses platform-appropriate TTS commands

Missing Dependency Validation in listen.py (v1)

- listen.py would crash with an ImportError stack trace if sounddevice was not installed
- The crash output would confuse agent.js's stdout parsing, causing incorrect fallback behavior
- Fix: Packages are imported inside a try/except block; failures print a structured TEXT: ERROR_ code

8. Setup & Run Guide

This section contains everything someone needs to run the project from scratch.

8.1 Prerequisites

Requirement	Status	Notes
Node.js v18+	Required	nodejs.org — verify with: node --version
Python 3.8+	Required	python.org — verify with: python3 --version
Google Chrome	Required	google.com/chrome — install in default OS location
Ollama	Optional	ollama.com — needed for AI form filling
Llama 3.2 model	Optional	Run: ollama pull llama3.2 (after installing Ollama)
Python packages	Optional	pip install SpeechRecognition sounddevice numpy
espeak	Linux only	sudo apt install espeak (for TTS on Linux)

Requirement	Status	Notes
portaudio19-dev	Linux only	sudo apt install portaudio19-dev (before pip packages)

8.2 Run Steps

1. Unzip the project folder
2. Open a terminal inside the folder and run: npm install
3. (Optional) In a separate terminal, start Ollama: ollama serve
4. **Run the agent: node agent.js**
5. Follow prompts — answer by voice or typing

8.3 Troubleshooting

Problem	Solution
Chrome not found	Install Google Chrome in the default OS location
Ollama offline message	Run ollama serve in a separate terminal. Agent still works without it.
Voice not working	Install Python packages (pip install speechrecognition sounddevice numpy). Agent falls back to typed input.
npm install fails	Check Node.js version: node --version must be 18 or higher
Linux audio errors	Run: sudo apt install portaudio19-dev, then reinstall Python packages

9. Security & Privacy Notes

The following files are generated at runtime and contain sensitive data. They must never be shared or committed to version control:

File	Sensitive Data
user_profiles.json	Contains real name, username, date of birth, country, gender, master password
accounts_db.json	Contains created account emails and plaintext passwords per website

The v2 release ships with a blank user_profiles.json. A .gitignore file is recommended to exclude both data files if the project is version controlled.

The agent sets a realistic Windows Chrome user agent and uses puppeteer-extra-plugin-stealth to avoid bot detection. However, automated account creation may violate the Terms of Service

of many websites. This tool should only be used on websites you own or have explicit permission to test.

10. Summary

Attribute	Value
Project Name	temp-mail-agent
Version	2.0.0
Language	JavaScript (Node.js ESM) + Python 3
AI Model	Llama 3.2 via Ollama (local, offline-capable)
Browser Engine	Puppeteer v24 + Stealth plugin
Platform Support	Windows 10/11, macOS 12+, Linux (Ubuntu 20.04+)
Voice I/O	Python SpeechRecognition + OS-native TTS
Temp Email	1secmail API (primary), Temp-Mail.org (fallback)
Credential Storage	accounts_db.json (local, plaintext — keep private)
Critical Bugs Fixed	Duplicate const page, hardcoded macOS paths, missing dependency handling

End of Report — temp-mail-agent v2.0 — April 2026