# Autonomous liquid-handling robotics scripting through large language models enables accessible and safe protein engineering workflows

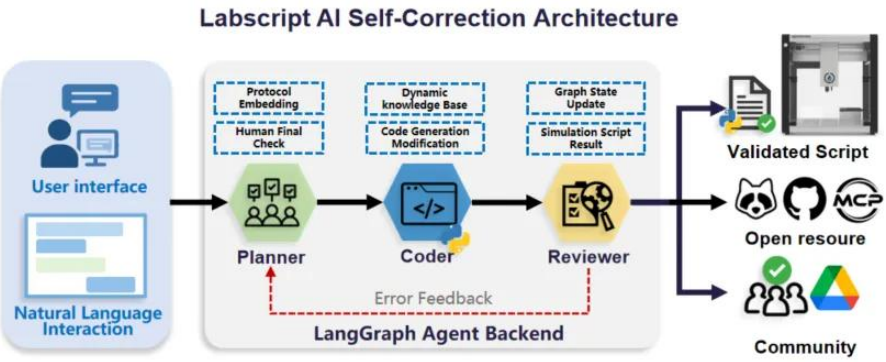
Yuan Gao#, Yizhou Luo#, Yunquan Lan, Han Jiang, Lihao Fu*, Tong Si*

State Key Laboratory for Quantitative Synthetic Biology, Shenzhen Institute of Synthetic Biology, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China

* To whom correspondence should be addressed
tong.si@siat.ac.cn
lh.fu@siat.ac.cn

**Abstract**

Laboratory automation enhances experimental throughput and reproducibility in synthetic biology, yet widespread adoption remains limited by the programming expertise required to operate robotic platforms. Large Language Models (LLMs) offer promising solutions but lack domain-specific knowledge to program robotics for reliable synthetic biology workflows. We developed LabscriptAI, a multi-agent framework that enables LLMs to autonomously generate and validate executable Python scripts for customized hardware configurations through iterative self-correction. We demonstrate LabscriptAI capabilities through proof-of-concept adapting a community protocol for Plate Reader Fluorescence Calibration, followed by validation using cell-free protein synthesis (CFPS) workflows encompassing expression, purification, and quantification of green fluorescent protein (GFP). For real-world protein engineering, we first applied LabscriptAI to the Critical Assessment of Protein Engineering (CAPE) competition, automating fluorescence characterization of >300 unique GFP variants designed by 53 student teams from 5 countries. Furthermore, LabscriptAI orchestrated distributed automation across a biofoundry work cell and fume hood-enclosed liquid handling robot to screen formaldehyde-converting enzyme variants. This integrated workflow successfully identified improved double mutants, including variants with more than 3-fold enhanced catalytic efficiency and improved substrate tolerance. The framework provides an intuitive natural language interface for generating robust automation protocols, facilitating human-robot collaboration by engaging research communities on the cloud and enhancing laboratory safety through flexible automation. Additionally, LabscriptAI autonomously interfaces with community databases following established standards to retrieve and deposit standardized data and protocols. Overall, our approach democratizes laboratory automation by eliminating programming barriers while ensuring reproducible, community-driven protocol development for synthetic biology research.



## 1. Introduction

Laboratory automation has become essential for enhancing experimental throughput, reproducibility, and efficiency in synthetic biology research[1, 2]. Sophisticated biofoundries provide centralized high-throughput robotics and standardized workflows for large-scale projects, while accessible platforms like Opentrons have democratized laboratory automation through affordability and open-source design[3]. For example, the Opentrons community has developed extensive open-source protocols for molecular biology workflows, with projects like AssemblyTron demonstrating flexible DNA assembly automation on OT-2 systems.[4]

Despite these advances, scripting liquid handling robots for customized workflows remains a persistent bottleneck requiring substantial programming expertise[5]. While community protocols provide standardized solutions, adapting protocols to specific hardware configurations or experimental requirements demands extensive coding knowledge. This challenge is compounded by the diversity of robotic platforms across the field. For example, the Opentrons ecosystem includes multiple models—the standard OT-2 and advanced OT-Flex[6]—with varying deck configurations and modules that necessitate non-trivial code modifications when transitioning between systems. In addition to Opentrons, researchers employ a range of more advanced liquid handling platforms including Tecan Evo and Fluent systems, Hamilton STAR and Vantage series, and others, each with proprietary programming interfaces and distinct operational paradigms. Commercial platforms typically employ closed-source software environments that further complicate cross-platform protocol adaptation. This fragmented landscape creates substantial programming barriers that prevent synthetic biologists from fully utilizing automation capabilities despite growing institutional investment and community support.

Given these barriers, Large Language Models (LLMs) present transformative opportunities to bridge this accessibility gap. Inagaki and colleagues demonstrated that GPT-4 could generate functional OT-2 scripts from natural language using simple feedback loops with simulators, though this approach struggled with complex protocols and sometimes circumvented errors rather than implementing correct experimental logic[7]. Hardware-agnostic interfaces like PyLabRobot have simplified multi-platform control[8], yet their integration with LLMs remains limited. The reliable application of LLMs to synthetic biology automation is hindered by several fundamental challenges: (1) general-purpose LLMs lack domain-specific knowledge of synthetic biology protocols, reagent handling requirements, and safety constraints; (2) LLMs struggle with hardware-specific constraints including pipetting limitations, add-on modules, timing dependencies, and deck layout restrictions that vary substantially across platforms; (3) complex multi-step workflows require sophisticated fine-tuning and validation, but existing approaches lack rapid iteration cycles of script generation and debugging by software simulation before committing to actual experimental execution; and

(4) developing reproducible, community-standard protocols demands integration with established databases and metadata standards—capabilities absent from existing LLM-based automation approaches. These limitations prevent the deployment of LLM-generated scripts in real-world synthetic biology applications where accuracy, safety, and reproducibility are paramount.

Here, we present LabscriptAI, an open-source framework that addresses these limitations through autonomous code generation and iterative refinement. Building on CodeAct as the core framework[9], our approach enables LLMs to write executable Python code as actions within a structured multi-agent system. Rather than generating static scripts, LLMs propose actions, execute them through Python interpreters, and analyze outcomes—including errors—to iteratively refine protocols. When errors occur, a Precise Refactoring Engine (PRE) generates targeted modifications to fix specific issues rather than regenerating entire scripts, achieving higher accuracy and faster convergence to validated protocols. PRE proves particularly effective for adapting existing scripts to new hardware configurations or experimental requirements. To enrich domain-expertise guidance, LabscriptAI integrates a dynamic knowledge base that captures validated protocols, hardware specifications, and controlled terminology, while a human-in-the-loop strategy ensures synchronized updates to both natural language standard operation procedures (SOPs) and corresponding Python robotic scripts for reliable intent interpretation. LabscriptAI implementation was validated through a series of tasks: proof-of-concept pattern generation on microtiter plates, cell-free protein synthesis, standardized characterization of 318 GFP variants for the Critical Assessment of Protein Engineering (CAPE) competition, and enzyme variant screening involving the safe handling of toxic substrates.

## 2. Materials and Methods

### 2.1 Hardware configuration

For the Opentrons liquid handling robots, the OT-2 model was equipped with 1-channel and 8-channel pipettes, along with a Temperature Module and a Magnetic Module. The OT-Flex model was configured with 1-channel and 8-channel pipettes, a Thermocycler Module GEN2, a Heater-Shaker Module, a Magnetic Block, and a Temperature Module, along with the integrated Flex Gripper for labware movement.

The Hamilton Vantage was configured with an 8-channel pipetting arm, a Heater-Shaker Module for incubation and agitation, an integrated Gripper for labware movement, and dedicated carriers for pipette tip racks and reagent reservoirs.

The biofoundry work cell configuration was described previously.[10]

## 2.2 Software Framework

LabscriptAI was implemented as a multi-agent system orchestrated using LangGraph (v0.4.8)[11]. The web-based user interface was developed using React (v18.3.1) and deployed at https://ai4ot.cn/. The system architecture comprised a Task Planning Agent that coordinates workflow processing and initiates human-in-the-loop dialogue to validate natural language protocols. The Code Graph backend contains two primary agents: a **Coder** agent for generating Python scripts using platform-specific APIs, and a **Reviewer** agent for code validation using appropriate simulators. When validation errors were detected, the **Reviewer** passed error messages back to the **Coder** agent, which employed the Precise Refactoring Engine (PRE) to generate SEARCH and REPLACE instructions targeting specific error locations rather than regenerating entire scripts. The **Reviewer** agent executed these modifications and re-validated the updated code. This Coder-Reviewer interaction cycle continued iteratively, with a maximum limit of nine iterations per script generation request.

Platform-specific simulators were integrated for validation: opentrons_simulate (v8.5.1) for Opentrons systems, the PyHamilton module of PyLabRobot (v0.1.6) for Hamilton STAR, and the newly developed pyFluent for Tecan Fluent to provide a standardized Python interface compatible with proprietary Tecan software environment.

## 2.3 LLM Selection and Prompt Engineering

The framework integrated Gemini 2.5 Pro through the OpenRouter API (v1.0) for all LLM operations. Prompts were structured with three hierarchical components: (1) system-level instructions defining agent roles and operational constraints, (2) hardware constraints from the static configuration files VALID_LABWARE_NAMES and VALID_INSTRUMENT_NAMES, and (3) user-provided experimental descriptions and parameters. To ground the LLM output and improve accuracy, dynamic retrieval and injection were performed to enrich prompts based on a growing knowledge base of CODE_EXAMPLES, which was constructed using validated scripts, embedded using Qwen3-Embedding-8B, and stored in Chroma DB.

## 2.4 Experimental

### Strains and Cultivation Conditions

*E. coli* DH5α (Sangon Biotech, Shanghai, China) was used for plasmid construction and amplification. *E. coli* BL21(DE3) (Sangon Biotech) served as the recombinant host for protein expression and formaldehyde FALD biotransformation. *E. coli* cells were cultured at 37°C with 220 rpm shaking in Luria-broth (LB) media supplemented with appropriate antibiotics (50 μg $mL^{-1}$ kanamycin (Kan) and/or 50 μg $mL^{-1}$

spectinomycin (Spec) for plasmid maintenance. Solid media were prepared by adding 1.5% (w/v) agar. All chemicals were purchased from Sigma–Aldrich (St. Louis, MO, USA) or Thermo Fisher Scientific (Pittsburgh, PA, USA) unless otherwise noted.

**DNA and Strain Construction**

All strains and plasmids used in this study are listed in Tables S1 and S2. Primers (Table S3) and genes were synthesized by GenScript Biotech (Nanjing, China). Enzymes for recombinant DNA cloning, including Q5 PCR polymerase, restriction endonucleases, and T4 ligase, were purchased from New England Biolabs (Ipswich, MA, USA). QIAprep Spin Plasmid Mini-prep Kits (Qiagen, Valencia, CA, USA) were utilized to isolate plasmid DNA from *E. coli*. PCR, digestion, and ligation products were purified using QIAquick PCR Purification and Gel Extraction Kits (Qiagen). Plasmid construction was performed via Gibson assembly using NEBuilder HiFi DNA Assembly Master Mix. Chemically competent *E. coli* was transformed using heat shock at 42°C. Sanger DNA sequencing was performed at Sangon Biotech.

**Cell-free GFP expression**

For NEB cell-free expression, reactions were assembled using the NEBExpress Cell-free E. coli Protein Synthesis System (New England Biolabs). Reaction mixtures contained 12 μL of S30 Synthesis Extract, 25 μL of 2X Protein Synthesis Buffer, 1 μL of T7 RNA polymerase, 1 μL of RNase Inhibitor (Murine), and template DNA (pET28-GFP variants) at 250 ng per 50 μL reaction. Reactions were brought to a final volume of 50 μL with nuclease-free water and incubated at 37°C with vigorous shaking for 4 hours. For high-throughput screening, reactions were assembled in 96-well plates using the automated liquid handling system with reaction volumes scaled to 25 μL.

**Bradford assay**

Protein concentration was determined using the Bradford Protein Assay Kit (P0006, Beyotime Biotechnology, Shanghai, China). A standard curve was generated using bovine serum albumin (BSA) diluted in PBS to concentrations ranging from 0 to 1.5 mg/mL. For the assay, 5 μL of each standard or sample was mixed with 250 μL of G250 dye reagent in a 96-well microplate. Following 10minutes incubation at room temperature, absorbance was measured at 595 nm using a microplate reader, and protein concentrations were calculated by comparison to the BSA standard curve.

**Manual GFP expression and purification**

Transformed BL21(DE3) cells were cultured in 1 L of LB medium at 37°C until OD600 reached 0.6-0.8, followed by inducible protein expression with 0.5 mM IPTG at 18°C for 16-20 hours. Cells were harvested

by centrifugation and resuspended in lysis buffer (50 mM Tris-HCl, 500 mM NaCl, 10 mM imidazole, pH 8.0). Cell lysis was performed by sonication on ice, cleared lysate by centrifugation (4,000 × g, 30 min, 4°C) was applied to Ni-NTA agarose (Qiagen), and proteins were eluted with 250 mM imidazole. Following concentration using Amicon Ultra-15 centrifugal filters (30 kDa MWCO, Millipore), proteins were further purified by size exclusion chromatography on a Superdex 200 pg 16/600 column using an ÄKTA prime PLUS system (GE Healthcare) equilibrated with 25 mM Tris-HCl, 150 mM NaCl, pH 8.0 and monitored for absorbance at 280 nm (A280).

**GFP characterization**

Fluorescence measurements were performed using an Agilent Microplate Reader with excitation at 488 nm and emission at 525 nm. For spectroscopic characterization of purified proteins, excitation spectra were recorded from 300-520 nm (λem = 550 nm) and emission spectra from 460-650 nm (λex = 430 nm), both with 2 nm bandwidth. Quantum yields were determined using avGFP as a reference standard (QY = 0.77) with integrated emission normalized to absorbance at 280 nm, according to the following equation,

$$QY_i = \frac{\int_{460nm}^{650nm}(F_i)d\lambda}{A280_{avGFP}} \frac{A280_{avGFP}}{\int_{460nm}^{650nm}(F_{avGFP})d\lambda} QY_{avGFP}$$

where QY denotes quantum yield, F denotes fluorescence emission intensity, i is different GFP, and A280 denotes absorbance at 280 nm.

**Construction of GALS Mutants**

GALS mutagenesis libraries were created by introducing specific mutations at select positions via PCR using appropriate primers (Table S3). Gibson assembly was used to construct plasmid libraries, which were used to transform *E. coli* DH5α on an automatic PCR machine. For each residue, 192 clones were randomly picked using a QPix colony picker for sequence verification. A representative collection of mutant strains covering all 19 possible amino acid substitutions were then selected by hit picking for plasmid DNA extraction using a liquid-handling robot with a magnetic stand and a centrifuge using N96 Mag-MK Plasmid DNA Mini-Preps Kit (TianGen Biotech Co., Ltd., Beijing, China). Plasmid DNA was used to transform BL21(DE3)-Δ6, and two random clones were selected for subsequent phenotypic screening.

**Whole-cell FALD Biotransformation for Mutant Screening**

Mutant colonies were picked into 96 deep-well microplates (Jetbiofil DMP221196) containing 200 μL of LB+Kan and incubated for 24 hrs at 37°C. Cultures were diluted 1:100 into 1 mL of fresh LB+Kan containing 0.1 mM isopropyl β-D-1-thiogalactopyranoside (IPTG) and incubated at 37°C for 4 hrs followed

by 20 hrs at 30°C. Cells were harvested by centrifugation at 3,300×*g* for 5 min, washed twice using phosphate-buffered saline (PBS, 50 mM, pH 7.4), and resuspended in assay buffer (50 mM PBS, 5 mM $MgSO_4$, 0.5 mM thiamine pyrophosphate (TPP), 1.5 g $L^{-1}$ FALD, pH 7.4). After 30°C incubation for 24 hrs, supernatants were collected after centrifugation at 3,300×*g* for 5 min for analysis.

**Spectrometric Assays for GA and DHA Detection**

For spectrometric GALD detection, diphenylamine (DPA) reagent was prepared by dissolving 1.5 g diphenylamine in 100 mL of glacial acetic acid followed by addition of 1.5 mL of concentrated sulfuric acid. Then, sample or standard solutions (30 μL) were mixed with 150 μL of DPA reagent in 96-well PCR plates (Jetkeen PC-0200-9B), incubated at 90°C for 30 min, and endpoint absorbance was measured at 650 nm in 96-well microtiter plates (Corning 3599).

For spectrometric DHA detection, 90 μL of sample or standard solutions were dried at room temperature in 96-well microtiter plates under nitrogen in a chemical fume hood. Samples were reconstituted in 90 μL of 50 mM PBS (pH 7.4), followed by addition of 60 μL of Buffer A (0.2 mg $mL^{-1}$ galactose oxidase, 24 U $mL^{-1}$ horse radish peroxidase, 50 mM PBS, 5 mM $MgSO_4$, pH 7.4) and then 50 μL of Buffer B (3.2 mM ABTS, pH 7.4). Absorbance at 410 nm was monitored for 20 min.

**Manual GALS Protein Expression and Purification**

Genes encoding the parent and mutant GALS were assembled into pET28m expression vector via Gibson Assembly. *E. coli* BL21(DE3) cells carrying different recombinant plasmids were cultivated in 5 mL of LB+Kan at 37 °C for 16-20 hrs and subsequently scaled to 1 L cultures. When $OD_{600}$ reached 0.6~0.8, IPTG was added to a final concentration of 0.5 mM, followed by 18 °C induction for 20 hrs. Cells were harvested by centrifugation at 8,000×*g* and resuspended in 50 mL of lysis buffer (50 mM Tris-HCl, 5 mM MgSO4, pH 7.5).

Cell lysis was performed using a high-pressure homogenizer (JNBIO, China), and debris was removed by centrifugation at 10,000×*g* for 30 min at 4 °C. The clarified lysate was loaded onto a nickel affinity column (GE Healthcare), rinsed with 50 mL of wash buffer (25 mM Tris-HCl buffer, 5 mM MgSO4, and 20 mM imidazole, pH 7.5,) and then eluted with 20 mL of elution buffer (25 mM Tris-HCl, 5 mM MgSO4, and 250 mM imidazole, pH 7.5). Purified protein was concentrated and buffer-exchanged into 50 mM PBS with 5 mM MgSO4 (pH 7.4) using an Amicon Ultra centrifugal filter device (Millipore, USA, 30 kDa cutoff). Protein concentration was determined using the Bradford Protein Assay Kit (Beyotime, Shanghai, China).

**In Vitro Enzyme Kinetics Characterization**

Initial reaction rates were measured at 37°C in 50 mM PBS (pH 7.4) containing 5 mM $MgSO_4$ and 0.5 mM TPP upon addition of 1 mg $mL^{-1}$ purified enzyme. FALD substrate concentrations ranged from 2 to 1,500 mM. Kinetic parameters $k_{cat}$ and $K_m$ were estimated by measuring initial velocities of enzymatic reaction and curve-fitting according to the Michaelis-Menten equation or allosteric sigmoidal equation, using GraphPad Prism 9 software. All measurements were performed in triplicate.

## 3.Results

### 3.1 System Architecture and Design Principles

LabscriptAI comprises three integrated components: a web-based user interface, a code generation and self-correction backend, and a storage layer that contains both long-term knowledgebase and dynamic project-specific data (Figure 1).

The web interface enables users to specify experimental goals through natural language input and configure hardware settings. Real-time visualization components display the dynamic generation and modification of both standard operating procedures and executable scripts, allowing researchers to monitor the transformation from biological concepts to robotic instructions. Optionally, the system accepts pre-existing natural language protocols and robotic scripts, supporting both protocol adaptation and code refinement workflows.

The Task Planning Agent coordinates workflow processing based on user inputs. The agent initiates human-in-the-loop dialogue to validate and refine natural language protocols, ensuring experimental steps accurately reflect user intentions. Once protocols are finalized, the agent determines the appropriate code generation strategy: adaptation workflows for existing scripts requiring modification for new hardware configurations, or de novo generation that translates validated protocols into platform-specific instructions. This hierarchical processing ensures accurate translation of experimental goals before entering the core Code Graph backend.

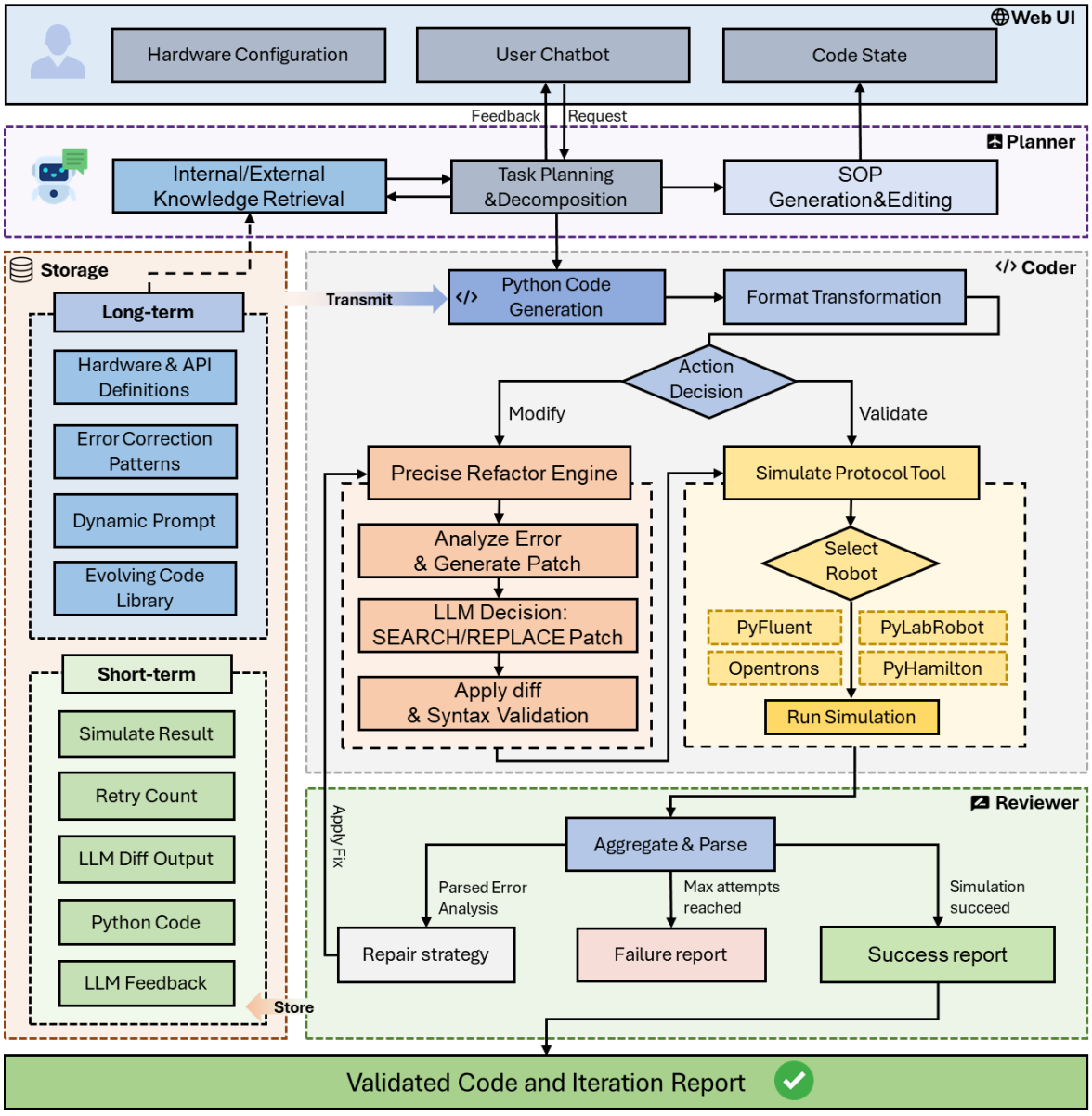



**Figure 1 LabscriptAI architecture**

The Code Graph backend orchestrates iterative code refinement through a multi-stage pipeline. Python script generation leverages knowledge base resources and project-specific context to produce initial implementations. Generated scripts undergo validation using platform-appropriate simulators that are selected based on target hardware. Successful simulations yield deployment-ready code, while validation failures activate the Code Modification Pipeline. The Precise Refactoring Engine analyzes error messages and generates targeted SEARCH/REPLACE operations, preserving validated code segments rather than regenerating complete scripts. This iterative cycle continues until validation succeeds or retry limits are reached.

Supporting this pipeline, the storage architecture separates persistent domain knowledge from dynamic project data. The knowledge base maintains validated protocols, hardware specifications, API definitions, and controlled terminology, while workspace storage manages session-specific data including simulation results and generation logs. Successful validations are promoted from workspace to knowledge base, creating a continuous learning mechanism that improves system performance over time.

These architectural elements combine to deliver three key advances: intelligent task planning that adapts to varying input scenarios, efficient code refinement through targeted patches, and continuous learning through systematic feedback integration. System transparency—from detailed iteration logs to real-time visualization—facilitates interactive debugging and user understanding, enabling researchers without programming expertise to generate robust automation protocols while ensuring accuracy through multi-stage validation.

### 3.2 Plate Reader Fluorescence Calibration using a Community Protocol

To establish basic functionality and demonstrate cross-platform compatibility, we selected the iGEM Plate Reader Fluorescence Calibration protocol as our proof-of-concept challenge. This protocol represents a critical foundation for quantitative fluorescence measurements in synthetic biology, enabling researchers to convert arbitrary fluorescence units to standardized fluorescein equivalents for reproducible data comparison across laboratories and instruments[12]. The iGEM Foundation has developed and maintained this protocol, hosting it on protocol.io as a community standard that exemplifies best practices in experimental documentation and version control. To integrate into this community effort, we implemented a specialized collaboration agent that autonomously interfaces with protocol.io, by retrieving the latest protocol version, and posting generated robotics scripts back through API.

The natural language input to LabscriptAI consisted of the request "Prepare a fluorescein standard curve following iGEM protocol" coupled with real-time protocol retrieval and parsing. The Task Planning Agent processed the protocol requirements and generated structured procedures comprising: (1) preparation of 10 μM fluorescein stock solution, (2) serial 2-fold dilutions across 7 concentration points, (3) triplicate plating in 96-well black plates, and (4) final volumes of 200 μL per well.

The Code Graph backend subsequently generated platform-specific scripts across four distinct liquid handling systems: Opentrons OT-2, Opentrons Flex, Tecan Fluent, and Hamilton STAR. For the Opentrons platforms, scripts utilized the native Python API[13]. For Hamilton STAR, the PyHamilton component of PyLabRobot was employed. For Tecan Fluent, as the PyLabRobot was compatible only with an older Tecan Evo model, we developed pyFluent to address the proprietary nature of Tecan software environment by

providing a standardized Python API. This development enables seamless script adaptation between platforms through the hardware-agnostic code generation approach of LabscriptAI.

The Precise Refactoring Engine proved particularly effective for cross-platform adaptation, generating targeted modifications for hardware-specific constraints including pipette tip compatibility, deck layout differences, and liquid handling protocols. Experimental validation demonstrated high precision and reproducibility across all platforms. Fluorescence measurements of the resulting standard curves showed coefficients of variation below 5% for triplicates, with linear regression $R^2$ values exceeding 0.999 for log-transformed data across all systems (Figure 2).

This validation establishes the capability of LabscriptAI to translate standardized biological protocols into executable, platform-optimized scripts while maintaining scientific rigor and reproducibility. The successful development of pyFluent and cross-platform protocol generation provide essential infrastructure for complex synthetic biology workflows requiring hardware flexibility and protocol standardization.

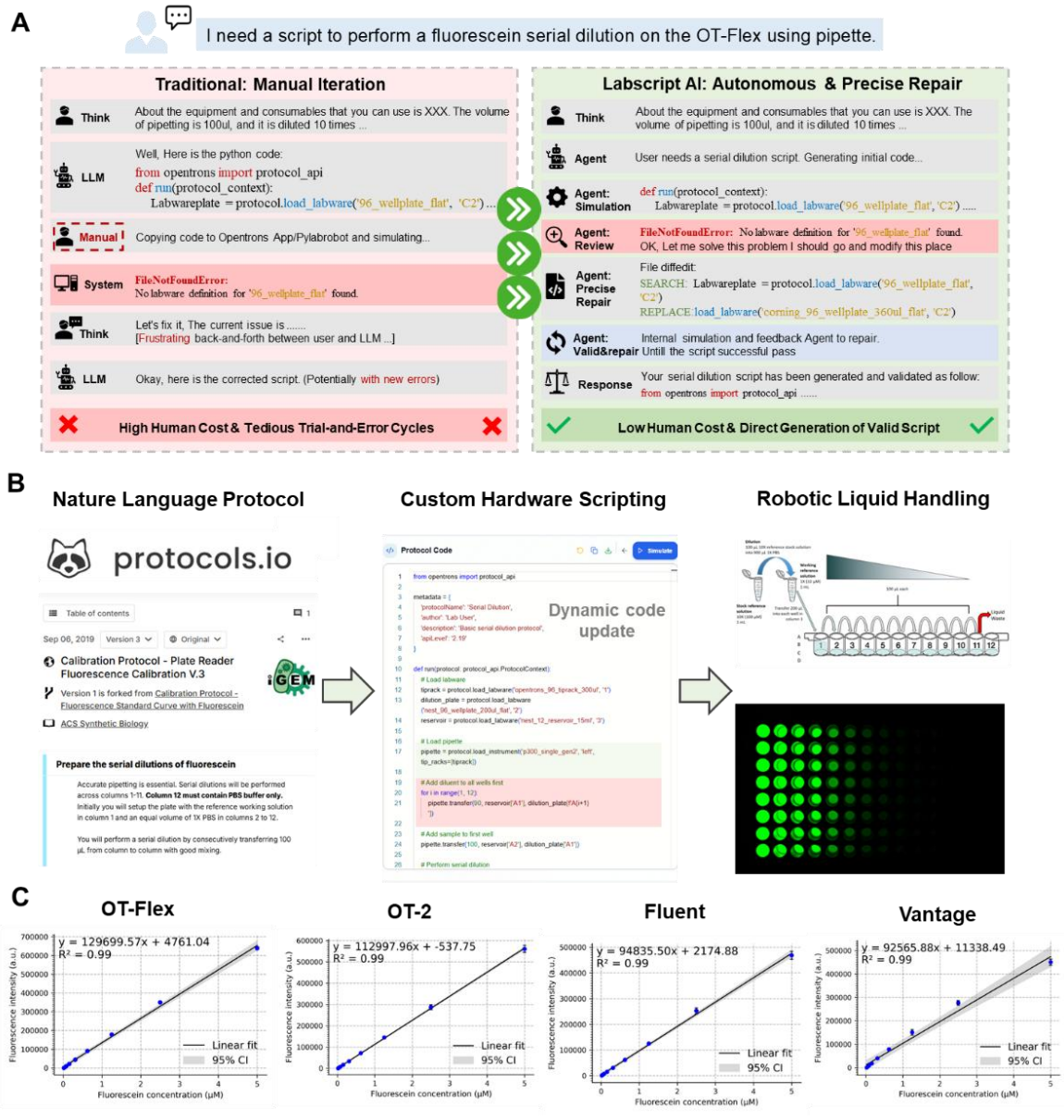



**Figure 2 Robotic scripting for Plate Reader Fluorescence Calibration using a Community Protocol**

### 3.3 Cell-Free GFP Expression for Student Competition on Protein Design

To validate the system in a biological context, we automated a CFPS workflow to express GFP using the pET28 vector on the Opentrons Flex. The natural language input requested: "Set up cell-free GFP expression reactions with varying template concentrations, include positive and negative controls" along with the NEB cell-free expression protocol. LabscriptAI generated a protocol that included: (1) preparation of CFPS master mix and aliquot into a 96-well plate, (2) template DNA dilution series, and (4) incubation

at 37°C for 4 hours. Initial validation with wild-type avGFP demonstrated robust and reproducible expression, and negative controls lacking template DNA showed negligible background fluorescence.

Encouraged by these results, LabscriptAI was applied to support the Critical Assessment of Protein Engineering (CAPE) competition. CAPE was established in 2023 to address the critical gap between computational protein design capabilities and experimental validation throughput.[14] Modeled after CASP, which drove breakthroughs in structure prediction through community-wide blind assessments, CAPE provides a platform where student teams develop computational models, submit designed sequences for centralized experimental validation, and receive standardized performance data. This framework accelerates learning cycles in protein engineering by providing high-quality, unbiased experimental data that teams can use to benchmark and refine their computational approaches[14].

The 2025 CAPE challenge focused on engineering GFP variants with enhanced brightness and thermostability. Teams received 50,000 sequence-function data points from previous GFP studies as training data, then designed 6 variant sequences each containing up to 6 mutations in the avGFP sequence. To accommodate the competition scale, LabscriptAI demonstrated its adaptation capabilities by modifying the validated CFPS protocol for wild-type avGFP. When prompted with "Adapt the GFP expression protocol for 318 samples across multiple plates" the system automatically: (1) reconfigured the workflow for four 96-well plates, (2) optimized master mix preparation volumes, and (3) scheduled sequential fluorescence measurements to maintain consistent timing.

LabscriptAI successfully automated the characterization of 318 dereplicated GFP variants from 53 participating teams from five countries (China, Singapore, Russia, Thailand, and Malaysia). Each variant underwent standardized expression conditions with fluorescence measurements before and after heat treatment, normalized against fluorescein calibration curves run with each batch. The best-performing variants achieved remarkable improvements over wild-type GFP. Analysis of winning designs revealed convergent strategies, with successful teams employing ensemble models and structure-informed machine learning approaches. Through automated high-throughput validation, CAPE2025 demonstrated how LabscriptAI transforms months of manual experimentation into days of automated workflows, accelerating the pace of protein engineering discovery.

### 3.4 Benchmark Community-Designed GFP Variants with Public Database using Purified Forms

To rigorously evaluate CAPE-winning GFP variants, we sought quantitative comparison against established benchmarks from FPBase, a community-curated database that catalogs photophysical properties of over 800 fluorescent proteins[15]. While CFPS provides valuable screening data, the measured fluorescence is

compounded by expression level, intrinsic brightness, and assay conditions[16]. Accurate assessment of photophysical improvements, including quantum yield, extinction coefficient, and molecular brightness, requires protein purification to eliminate expression variations. We therefore challenged LabscriptAI with the natural language request: "Quantitatively compare CAPE GFP variants with FPBase bright GFP variants in purified form."

To streamline interaction with external databases, a specialized LLM agent was introduced for FPBase integration. Through API, this agent autonomously retrieved and parsed database entries, identifying sfGFP as a high-performance comparator. The agent extracted key metric descriptions from FPBase and transmitted these specifications to LabscriptAI.

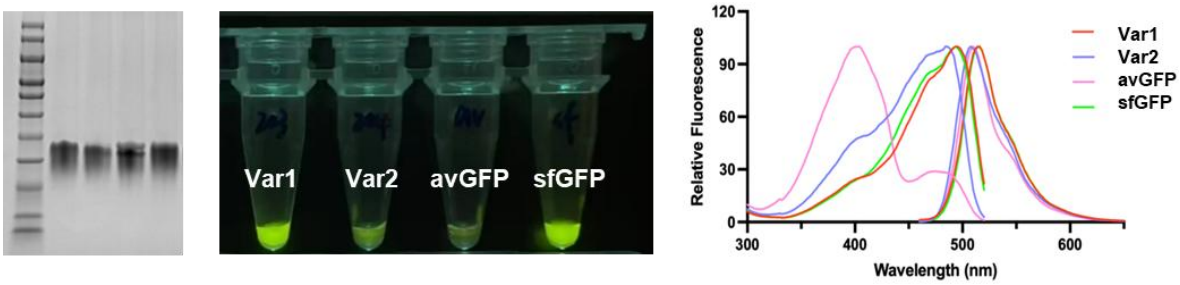



**Figure 3 GFP variants designed by student competition**

Through human-in-the-loop interaction, LabscriptAI generated an updated cell-free GFP expression SOP with four key protocol adaptations: (1) automated magnetic bead purification using Ni-NTA capture with stepwise imidazole washes and elution[17]; (2) Bradford assay quantification with 7-point BSA standard curves to instruct protein concentration normalization; (3) collection of full excitation and emission spectra for photophysical characterization; and (4) absorbance measurements for quantum yield calculation. The generated script orchestrated complex plate movements using the integrated gripper, transferring samples between Heater-Shaker modules for incubation, Magnetic Blocks for separation, and plate readers for spectroscopic analysis. However, SDS-PAGE analysis of the plate-based purification products revealed insufficient purity levels with substantial contaminating protein bands that would confound accurate photophysical characterization. Given these limitations, we updated our approach to employ conventional column chromatography for selected variants.

The top-performing CAPE variants, wild-type avGFP, and sfGFP were expressed in *E. coli* BL21(DE3) and manually purified using established protocols. Following IPTG induction, cells were lysed, and proteins were purified via Ni-NTA affinity chromatography followed by size exclusion chromatography. Spectroscopic characterization at normalized protein concentrations revealed that the best-performing

variant achieved 5.0-fold enhanced fluorescence compared to wild-type avGFP, reaching 80% of the level observed for the extensively optimized sfGFP benchmark (Figure 3).

The workflow concluded with automated data deposition back to FPBase through the collaborating agent. The system formatted characterization data according to FPBase standards, including excitation/emission spectra, quantum yields, brightness, and extinction coefficients. This bidirectional integration—where community databases provide both input standards and receive validated results—exemplifies how multi-agent AI systems can create self-improving scientific workflows that both leverage and contribute to collective knowledge.

### 3.5 Enzyme Variant Libraries Screening involving Hazardous Reagents

To extend LabscriptAI applications beyond fluorescent proteins, we evaluated its performance with enzyme engineering workflows involving hazardous reagents. While the CAPE competition demonstrated the utility of cloud laboratory settings for community-driven protein design, the present case targets one-carbon (C1) biomanufacturing to produce high carbon-number molecules using formaldehyde (FALD) substrate—a known carcinogen requiring specialized safety protocols. Robotic automation is particularly well-suited for such applications, as it mitigates safety risks to researchers associated with toxic substrate handling.

Variant libraries of glycolaldehyde synthase (GALS), an engineered enzyme from *Pseudomonas putida*, were selected for screening. GALS catalyzes a key step by condensing FALD into glycolaldehyde (GALD) and dihydroxyacetone (DHA).[18] We previously developed rapid spectrophotometric assays for detection and quantification of GALD and DHA products from whole-cell FALD biotransformation by GALS-expressing *E. coli*.[10] A biofoundry centered on a Tecan Fluent robot was previously used to create and screen single-site saturation mutagenesis GALS libraries in a regular laboratory environment,[10] which posed potential risks despite carefully designed safety protocols to minimize exposure. Here we challenge LabscriptAI with two key advances: (1) distribute the workflow among a biofoundry housed in regular laboratory environment for strain construction, fume hood-enclosed Opentrons OT-2 for FALD biotransformation and product quantification, and human researchers for manual material transfers; and (2) combinatorial mutagenesis at the V281 and F282 residues, where single mutations had previously failed to yield improved variants.

We provided LabscriptAI with the natural language request: "Create a combinatorial double-site mutagenesis library at positions V281 and F282 of GALS, then screen for enhanced formaldehyde biotransformation activity using distributed automation between biofoundry and fume hood systems" along with previously developed scripts for biofoundry operations. The framework successfully integrated

existing molecular biology protocols with newly generated safety-critical handling procedures. LabscriptAI orchestrated a complex multi-platform workflow that included: (1) automated library construction using the Tecan Fluent system with individually ordered primers containing specific mutations (19 amino acid substitutions at each position, yielding 38 single mutants and 361 double mutants); (2) clone picking and sequence verification through the integrated colony picker and liquid handling systems; (3) automated cell culture, protein induction, and cell washing protocols on the biofoundry platform; (4) transfer of washed, induced cells from the regular laboratory environment to the fume hood-enclosed Opentrons OT-2; (5) automated preparation and distribution of FALD-containing assay buffer (1.5 g $L^{-1}$ FALD in 50 mM PBS with cofactors) under containment, followed by static biotransformation reactions at room temperature for 24 h; and (6) supernatant collection and splitting for separate derivatization and spectrophotometric quantification of GALD and DHA. The generated protocols ensured that all FALD handling occurred within the enclosed system, eliminating direct human exposure to the carcinogenic substrate while maintaining the throughput necessary for comprehensive library screening.

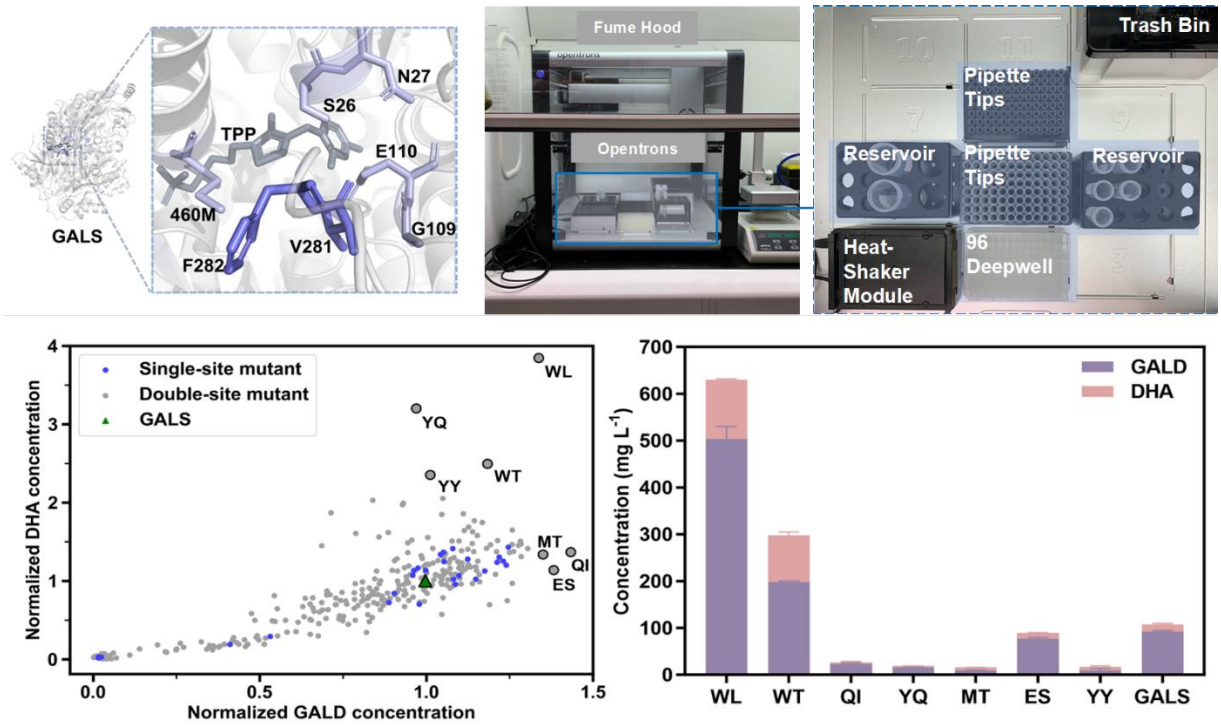



**Figure 4. Robotic GALS engineering**

The automated screening successfully identified several improved GALS variants with enhanced catalytic properties. Analysis of 399 screened variants revealed that several double mutants achieved improvements that were unattainable through single-site mutations at either position, thereby validating the combinatorial

mutagenesis strategy. Seven top variants (WL, YA, WT, YY, QI, MT, ES) were purified using a manual protocol to evaluate in vitro FALD transformation. Among these, WL and WT variants performed better than the parent GALS, while the remaining variants performed worse. The discrepancy between primary in vivo screening and secondary in vitro characterization had also been noted in previous GALS engineering studies (Figure 4). For the WL and WT variants, we further tasked LabscriptAI with developing automated workflows for enzymatic kinetics characterization. To promote data sharing and reproducibility, an LLM agent was employed to automatically interface with EnzymeML[19], and the raw data were processed using the community standard format and deposited along with associated metadata. Overall, the successful identification of enhanced GALS variants through this distributed automation approach demonstrates how LabscriptAI orchestrates complex, multi-platform workflows while ensuring researcher safety.

## Discussions

This study successfully demonstrated that LLM-driven automation can transform complex synthetic biology workflows from programming-intensive tasks into natural language-guided procedures. LabscriptAI achieved cross-platform protocol translation across distinct liquid handling systems while maintaining experimental precision. The framework enabled high-throughput characterization of 318 GFP variants for the CAPE competition, identifying variants with improved fluorescence compared to wild-type avGFP. In enzyme engineering applications, the system orchestrated distributed workflows between biofoundry equipment and fume hood-enclosed systems for enhanced researcher protection, successfully identifying GALS double mutants with 3.2-fold improved catalytic efficiency and enhanced substrate tolerance. These applications demonstrate that natural language interfaces can preserve the rigor of programmatic automation while promoting accessibility across synthetic biology workflows.

The multi-agent architecture proved particularly effective in managing experimental complexity while ensuring safety and reproducibility. The Task Planning Agent decomposed natural language requests into structured procedures, the Code Graph backend generated platform-specific implementations, and specialized collaboration agents interfaced with community databases for protocol retrieval and data deposition. This distributed intelligence enabled seamless handling of safety-critical workflows, as demonstrated in the GALS screening where formaldehyde handling was automated within containment while maintaining throughput for comprehensive library screening. The framework maintained experimental context across distributed platforms, coordinating timing constraints and resource allocation between instruments in separate laboratory zones or potentially operated by different research groups.

Integration with community infrastructure represents a transformative advance for reproducible synthetic biology through adherence to FAIR principles—Findable, Accessible, Interoperable, and Reusable[20-22].

LabscriptAI autonomously retrieved protocols from protocol.io, adapted them for specific hardware configurations, and deposited validated implementations back with comprehensive metadata. This bidirectional flow created a self-reinforcing ecosystem where successful protocols enhanced the knowledge base for future users. The framework interfaced with specialized databases—FPBase for fluorescent protein characterization and EnzymeML for enzyme kinetic data—ensuring that generated data followed community standards and remained accessible for meta-analyses. Dynamic retrieval-augmented generation from these curated resources provided the domain expertise that general-purpose LLMs lack, enabling accurate translation of biological requirements into robotic instructions.

Several design principles emerged as critical for successful implementation. Structured prompts focusing on biological and experimental parameters rather than programming details yielded the highest success rates, enabling researchers to bridge biological thinking and computational implementation without programming expertise. The Precise Refactoring Engine reduced convergence iterations compared to full script regeneration while mitigating LLM hallucination issues. The separation of persistent domain knowledge from dynamic project data in the storage architecture enabled continuous learning while maintaining validated protocol integrity, decreasing error rates relative to general-purpose LLM implementations. Human-in-the-loop interaction at critical decision points balanced automation with expert oversight, particularly important for protocols involving new experimental designs. Progressive validation from simulation to dry runs to wet runs ensured safe operation before deployment with valuable samples or hazardous materials.

Current limitations point toward specific development priorities. First, LabscriptAI currently focuses on scripting individual liquid-handling robots. LLM-guided scripting for robotic integration across multiple instruments including plate readers, incubators, and analytical devices would enable fully automated workflows. Second, complex procedures requiring real-time decision-making based on intermediate results remain challenging, as the framework operates on pre-defined protocols rather than adaptive experimentation. Hardware-in-the-loop feedback systems could enable real-time error detection and correction during execution, moving beyond the current reliance on pre-defined error patterns. Additionally, future development should incorporate experimental outcome data into the learning loop, enabling protocol optimization based on historical performance metrics, and integration with electronic laboratory notebooks for comprehensive experimental tracking.

## Conclusion

This study has demonstrated how LabscriptAI, built upon an iterative execution and self-correction engine, enables the adaptation and generation of complex synthetic biology workflows through natural language

instructions. The validated case studies—community protocol translation for fluorescence calibration, standardized cell-free screening of community-designed GFP variants, and distributed automation for hazardous enzyme library screening—showcase the flexibility, robustness, and accessibility of this approach. By eliminating programming barriers and enabling researchers to specify experimental goals in natural language, LabscriptAI democratizes laboratory automation while ensuring reproducibility through bidirectional integration with community databases and standardized metadata deposition. The open-source release of LabscriptAI provides the research community with a foundation for collaborative development, ensuring that advances in automation remain accessible to laboratories regardless of their computational resources or programming expertise. As LLMs and agentic frameworks continue to advance, the vision of a fully integrated, community-driven, and autonomous laboratory becomes increasingly feasible.

## References


(1) Hu, R.; Fu, L.; Chen, Y.; Chen, J.; Qiao, Y.; Si, T. Protein engineering via Bayesian optimization-guided evolutionary algorithm and robotic experiments. *Briefings in Bioinformatics* **2022**, *24* (1). DOI: 10.1093/bib/bbac570 (accessed 10/1/2025).

(2) Zhang, X.-E.; Liu, C.; Dai, J.; Yuan, Y.; Gao, C.; Feng, Y.; Wu, B.; Wei, P.; You, C.; Wang, X.; et al. Enabling technology and core theory of synthetic biology. *SCIENCE CHINA Life Sciences* **2023**, *66* (8), 1742–1785. DOI: https://doi.org/10.1007/s11427-022-2214-2.

(3) Chory, E. J.; Gretton, D. W.; DeBenedictis, E. A.; Esvelt, K. M. Enabling high-throughput biology with flexible open-source automation. *Molecular Systems Biology* **2021**, *17* (3), e9942. DOI: 10.15252/msb.20209942.

(4) Bryant, J. A.; Kellinger, M.; Longmire, C.; Miller, R.; Wright, R. C. AssemblyTron: flexible automation of DNA assembly with Opentrons OT-2 lab robots. *Synthetic Biology* **2023**, *8* (1), ysac032. DOI: 10.1093/synbio/ysac032.

(5) Laverick, A. OT-Mation: an open-source code for parsing CSV files into Python scripts for control of OT-2 liquid-handling robotics. *Synthetic Biology* **2025**, *10* (1), ysaf009.

(6) Opentrons Flex ™ Lab Robot Launches to Accelerate Bioautomation Across Thousands of Life Science Experiments. 2023.

(7) Inagaki, T.; Kato, A.; Takahashi, K.; Ozaki, H.; Kanda, G. N. LLMs can generate robotic scripts from goal-oriented instructions in biological laboratory automation. *arXiv* **2023**. DOI: 10.48550/arXiv.2304.10267.

(8) Wierenga, R. P.; Golas, S. M.; Ho, W.; Coley, C. W.; Esvelt, K. M. PyLabRobot: An open-source, hardware-agnostic interface for liquid-handling robots and accessories. *Device* **2023**, *1* (4), 100111. DOI: 10.1016/j.device.2023.100111.

(9) Wang, X.; Chen, Y.; Yuan, L.; Zhang, Y.; Li, Y.; Peng, H.; Ji, H. Executable Code Actions Elicit Better LLM Agents. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

(10) Luo, Y.; Fu, L.; Chen, J.; Xu, H.; Song, Z.; Jalal, K.; Chen, Y.; Xie, W.; Tian, S.; Fang, X.; et al. Mass Spectrometric Screening for Improving Enzymatic Conversion of Formaldehyde into C2 and C3 Products. *ACS Synthetic Biology* **2025**, *14* (7), 2718–2728. DOI: 10.1021/acssynbio.5c00151.

(11) LangGraph, T. LangGraph: Build resilient language agents as graphs. **2025**.

(12) Fedorec, A. J. H. FlopR: an open source software package for calibration and normalization of plate reader and flow cytometry data. *ACS Synthetic Biology* **2020**, *9* (9), 2258–2266.

(13) opentrons. Opentrons Protocol API v2 Documentation. **2025**.

(14) Fu, L.; Gao, Y.; Chen, Y.; Wang, Y.; Fang, X.; Tian, S.; Dong, H.; Zhang, Y.; Chen, Z.; Wang, Z.; et al. Critical Assessment of Protein Engineering (CAPE): A Student Challenge on the Cloud. *ACS Synthetic*

*Biology* **2024**, *13* (11), 3782–3787. DOI: 10.1021/acssynbio.4c00588.

(15) Lambert, T. J. FPbase: a community-editable fluorescent protein database. *Nature Methods* **2019**, *16* (4), 277–278. DOI: 10.1038/s41592-019-0352-8.

(16) Thornton, E. L. Cell-Free Protein Synthesis as a Method to Rapidly Screen Machine Learning-Generated Protease Variants. *ACS Synthetic Biology* **2025**, *14* (5), 1710–1718.

(17) *Protein quantification process using Opentrons Flex magnetic bead protein purification workstation*; Opentrons, 2025.

(18) Lu, X.; Liu, Y.; Yang, Y.; Wang, S.; Wang, Q.; Wang, X.; Yan, Z.; Cheng, J.; Liu, C.; Yang, X.; et al. Constructing a synthetic pathway for acetyl-coenzyme A from one-carbon through enzyme design. *Nature Communications* **2019**, *10* (1), 1378. DOI: 10.1038/s41467-019-09095-z.

(19) Lauterbach, S.; Dienhart, H.; Range, J.; Malzacher, S.; Spöring, J.-D.; Rother, D.; Pinto, M. F.; Martins, P.; Lagerman, C. E.; Bommarius, A. S.; et al. EnzymeML: seamless data flow and modeling of enzymatic data. *Nature Methods* **2023**, *20* (3), 400–402. DOI: 10.1038/s41592-022-01763-1.

(20) Torbiak, F. The Laboratory Automation Protocol (LAP) Repository. *ACS Synthetic Biology* **2023**.

(21) Morrell, W. T. The Experiment Data Depot: a web-based software tool for biological experimental data storage, sharing, and visualization. *ACS Synthetic Biology* **2017**, *6* (12), 2249–2252.

(22) Lister, A. Experimental Data Connector (XDC): Integrating the Capture of Experimental Data and Metadata Using Standard Formats and Digital Repositories. *ACS Synthetic Biology* **2023**, *12* (4), 1364–1370.