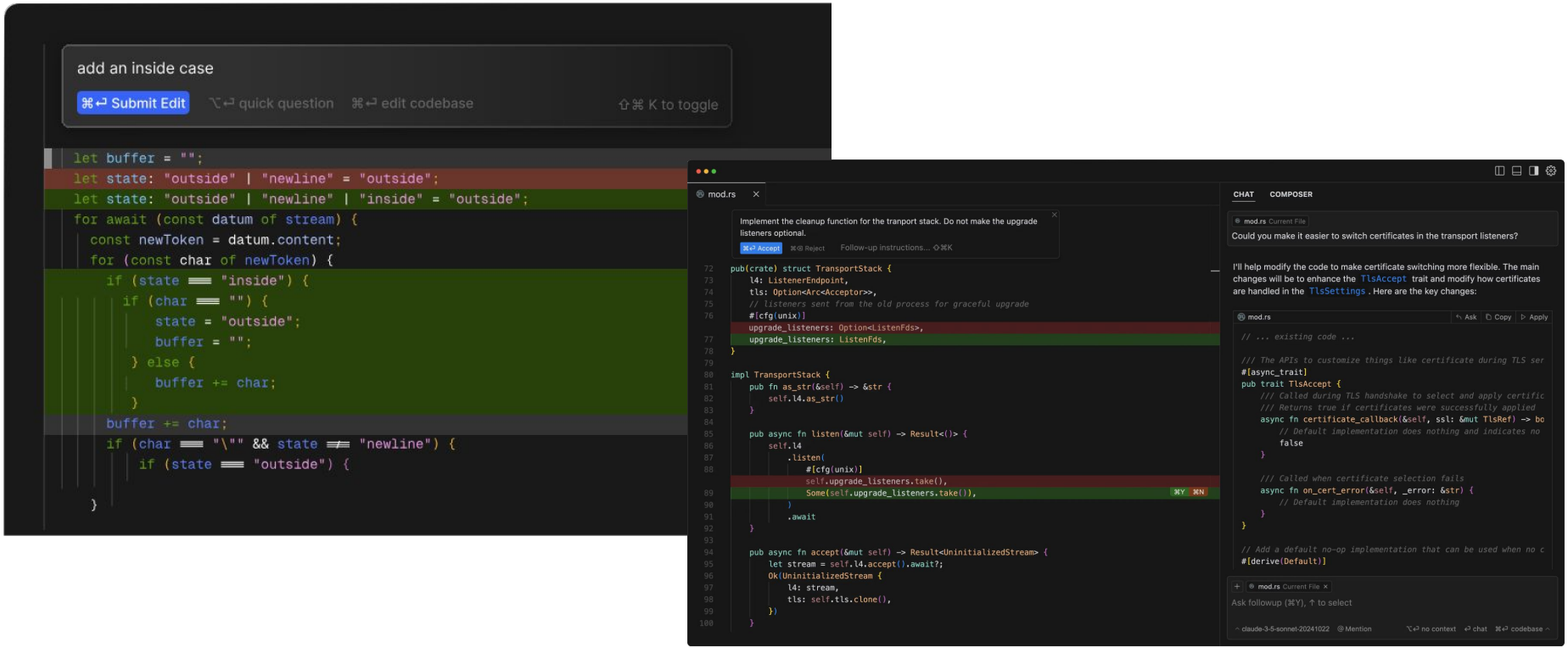
Bridging Design and Code with Gen AI

Lydia Chilton
7 April 2025

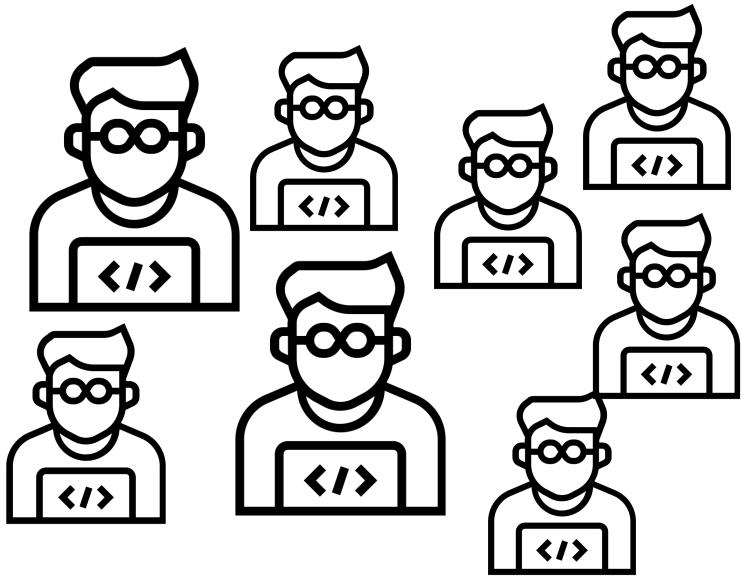
AI Code generation has transformed the way programmers work.



Design
(What do we build?)

Created by Brickclay
from Noun Project

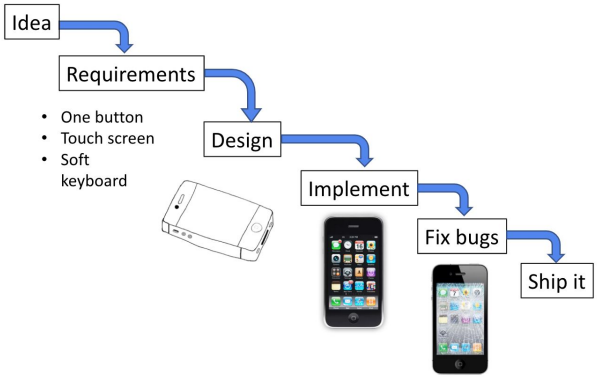
Engineering
(How do we build it?)



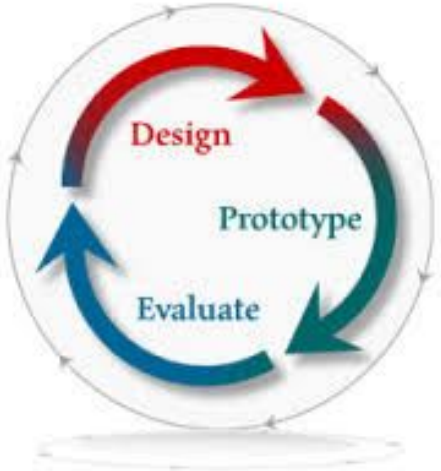
Created by WiStudio
from Noun Project

Two Software Design Processes

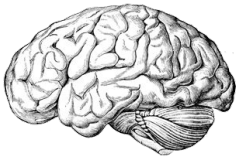
Feedforward



Feedback



The Waterfall Model

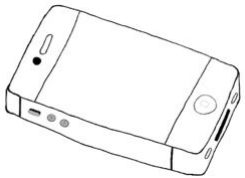


Idea

Requirements

- One button
- Touch screen
- Soft keyboard

Design



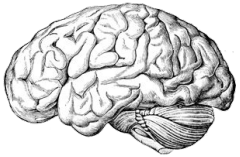
Implement

Fix bugs

Ship it

Product

The Waterfall Model: What could go wrong?



Idea

Requirements

- One button
- Touch screen
- Soft keyboard

Design

Implement

Fix bugs

Ship it

Product

What if a touch screen can't be implemented?

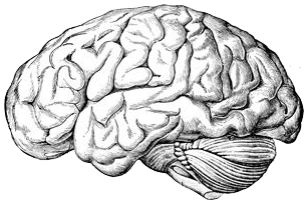
What if the hardware weighs 30 lbs?

What if this device is so slow it's unusable?

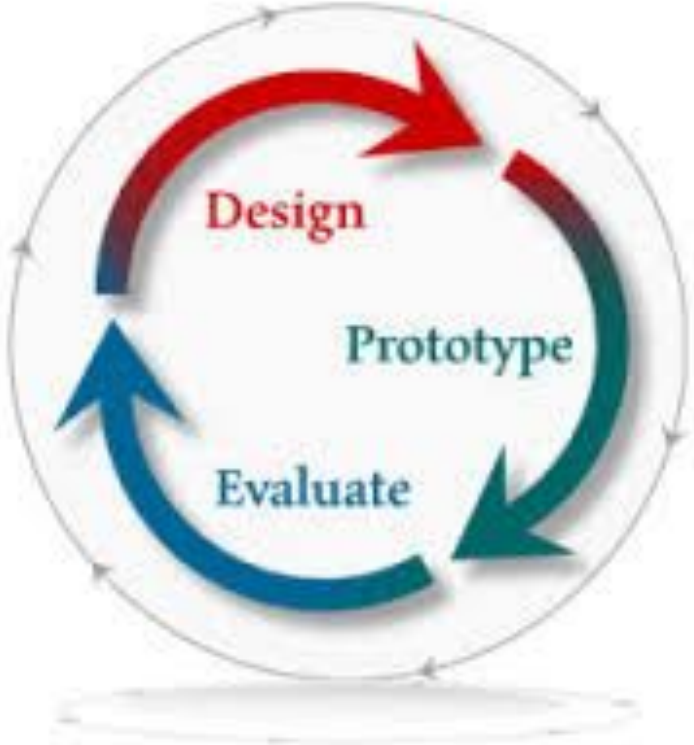
How can we keep up with the competition?

It becomes a feedback process anyway!

Iterative Design is a feedback loop.

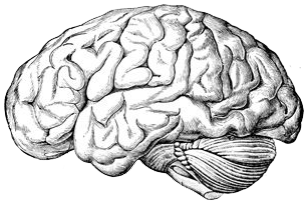


Idea

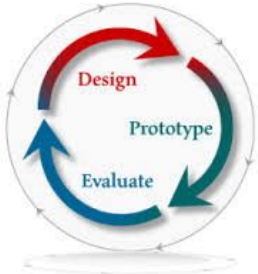


Product

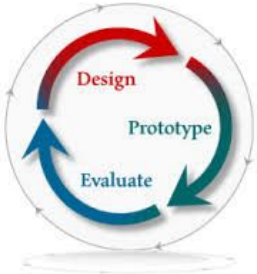
Each feature needs to be iterated before moving on



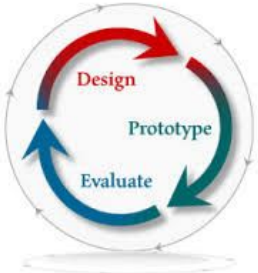
Idea



Touch screen



Soft keyboard

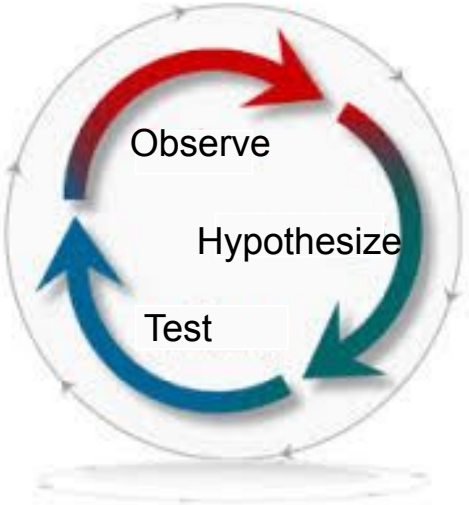


One button

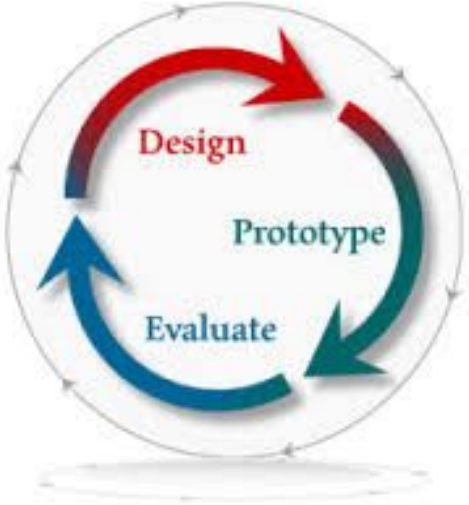
Product

Human-Centered Design: Two feedback loops

Person with a challenge



Ideas



Product

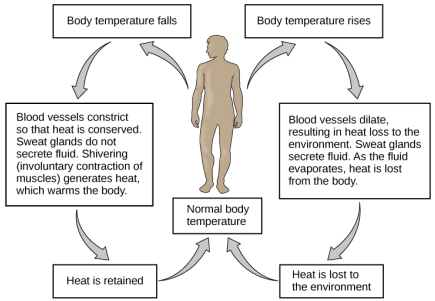
Understand the problem

Solve the problem

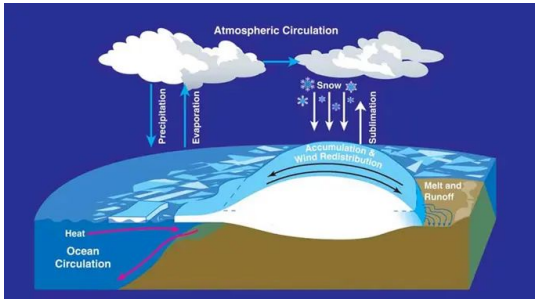
AI Code generation isn’t enough to create better software products…

We have to close feedback loops between design and engineering.

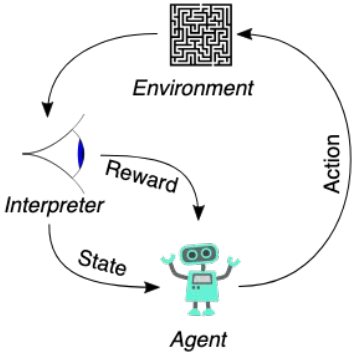
Feedback loops are essential to all systems



The human body



Earth systems

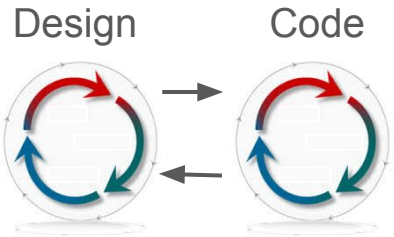


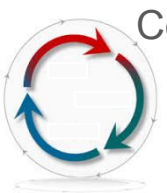
Training AI

Education

Childhood Development

Bridging Design and Code with Gen AI





Dynex

- Iterative Problem Specification
- Iterative Development

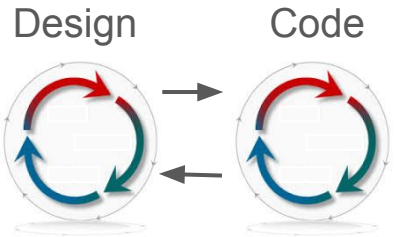
LogoMotion

- Self-debugging loop
- Iteration with AI Editing widgets

Double Agents

- Testing with user simulation
- Iterating on designs with AI suggestions

Bridging Design and Code with Gen AI



Dynex

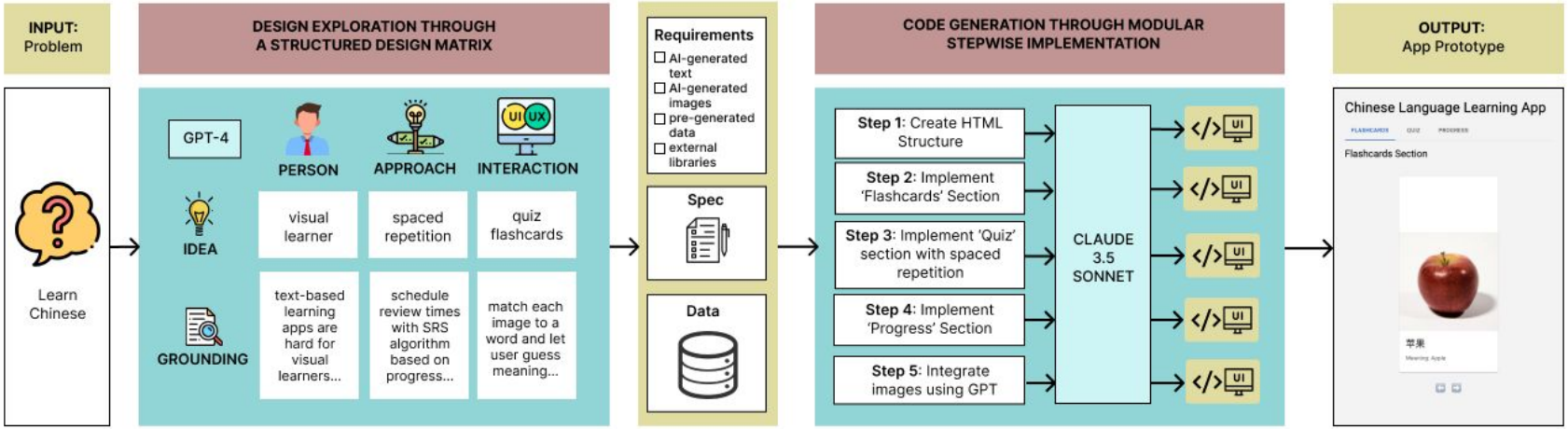
- Iterative Problem Specification
- Iterative Development

LogoMotion

- Self-debugging loop
- Iteration with AI Editing widgets

Double Agents

- Testing with user simulation
- Iterating on designs with AI suggestions



DynEx: Agentic Assistance to Bridge Design and Code

Jenny Ma, Karthik Sreedhar, Vivian Liu, Pedro Alejandro Perez, Sitong Wang, Riya Sahni, Lydia B. Chilton
CHI 2025

Everyone wants Agents to Develop Software

Published as a conference paper at ICLR 2024

SWE-BENCH: CAN LANGUAGE MODELS RESOLVE REAL-WORLD GITHUB ISSUES?

Carlos E. Jimenez[*1,2] **John Yang**[*1,2] **Alexander Wettig**[1,2]
Shunyu Yao[1,2] **Kexin Pei**[3] **Ofir Press**[1,2] **Karthik Narasimhan**[1,2]
[1]Princeton University [2]Princeton Language and Intelligence [3]University of Chicago

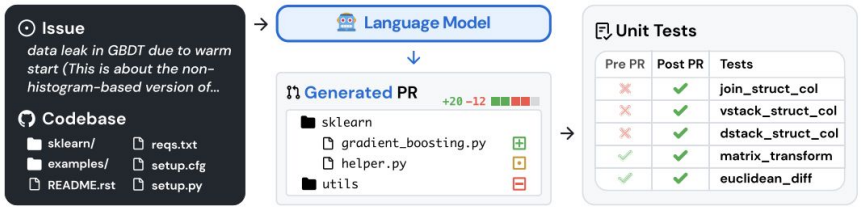
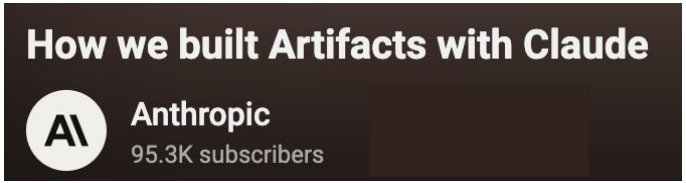
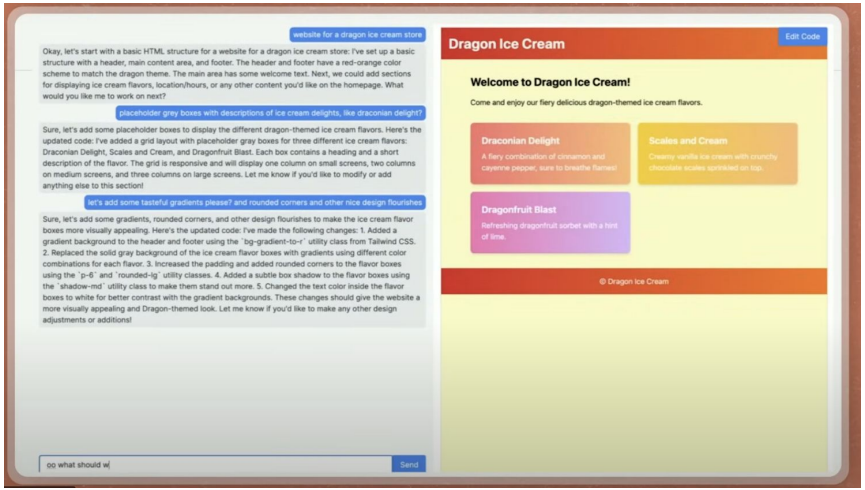


Figure 1: SWE-bench sources task instances from real-world Python repositories by connecting GitHub issues to merged pull request solutions that resolve related tests. Provided with the issue text and a codebase snapshot, models generate a patch that is evaluated against real tests.

SWE-Bench (11/2024)
https://arxiv.org/pdf/2407.16741





How we build Artifacts in Claude (8/2024)
https://www.youtube.com/watch?v=vUdNaAAc4FY

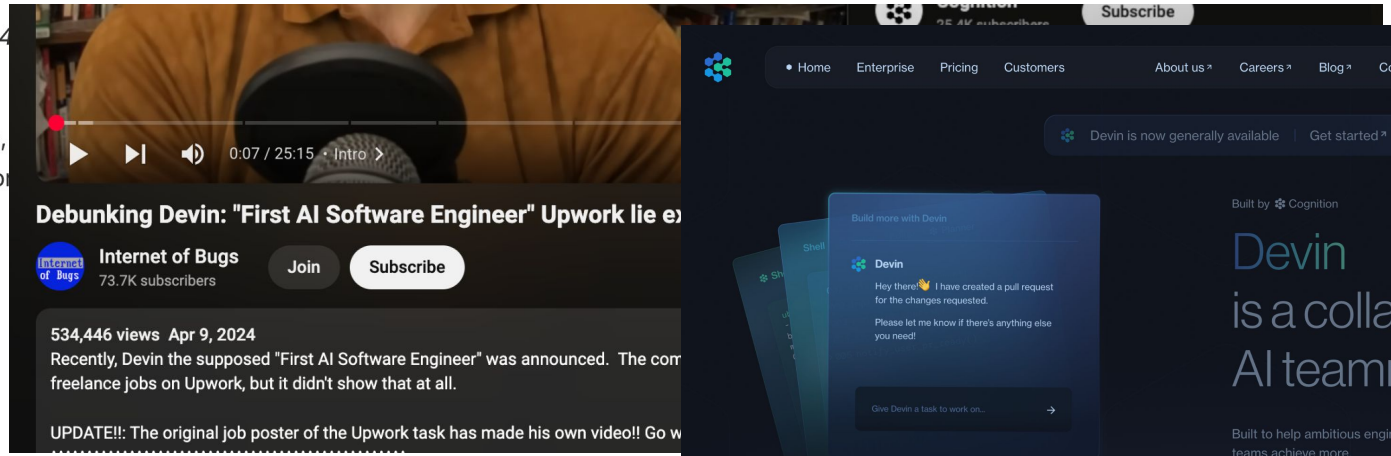
Can Agents do it by themselves? **NO!**

/blog

Introducing Devin, the first AI software engineer

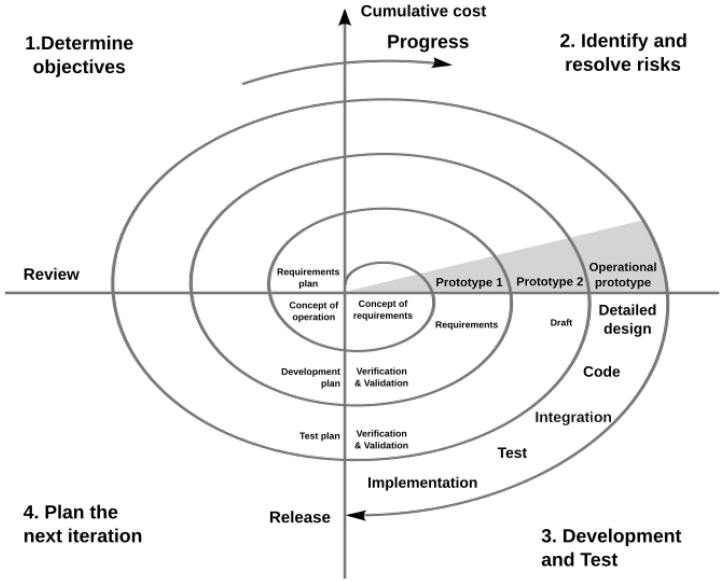
March 12, 2024

Devin is a tireless,
complete tasks fo



Why do we need human interaction?

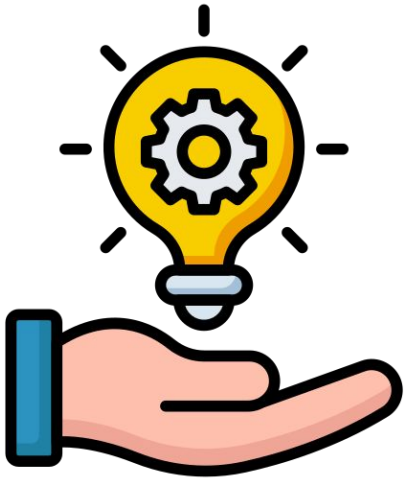
- Specify the problem and the solution
- Iteratively test modular pieces (or prototypes) so errors don’t compound.
  - Every problem has unknown aspects of the environment where it will operate.
  - We have to expect to find new problems.



We need to bridge design and implementation

There is a gap in going from an idea to a working solution

idea



solution

We need to specific the problem! Who is the user?

idea

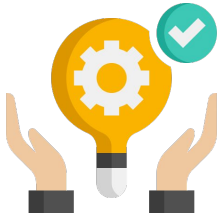
Who is the target user?

solution

We need to specific the problem! What is the **approach**?

idea

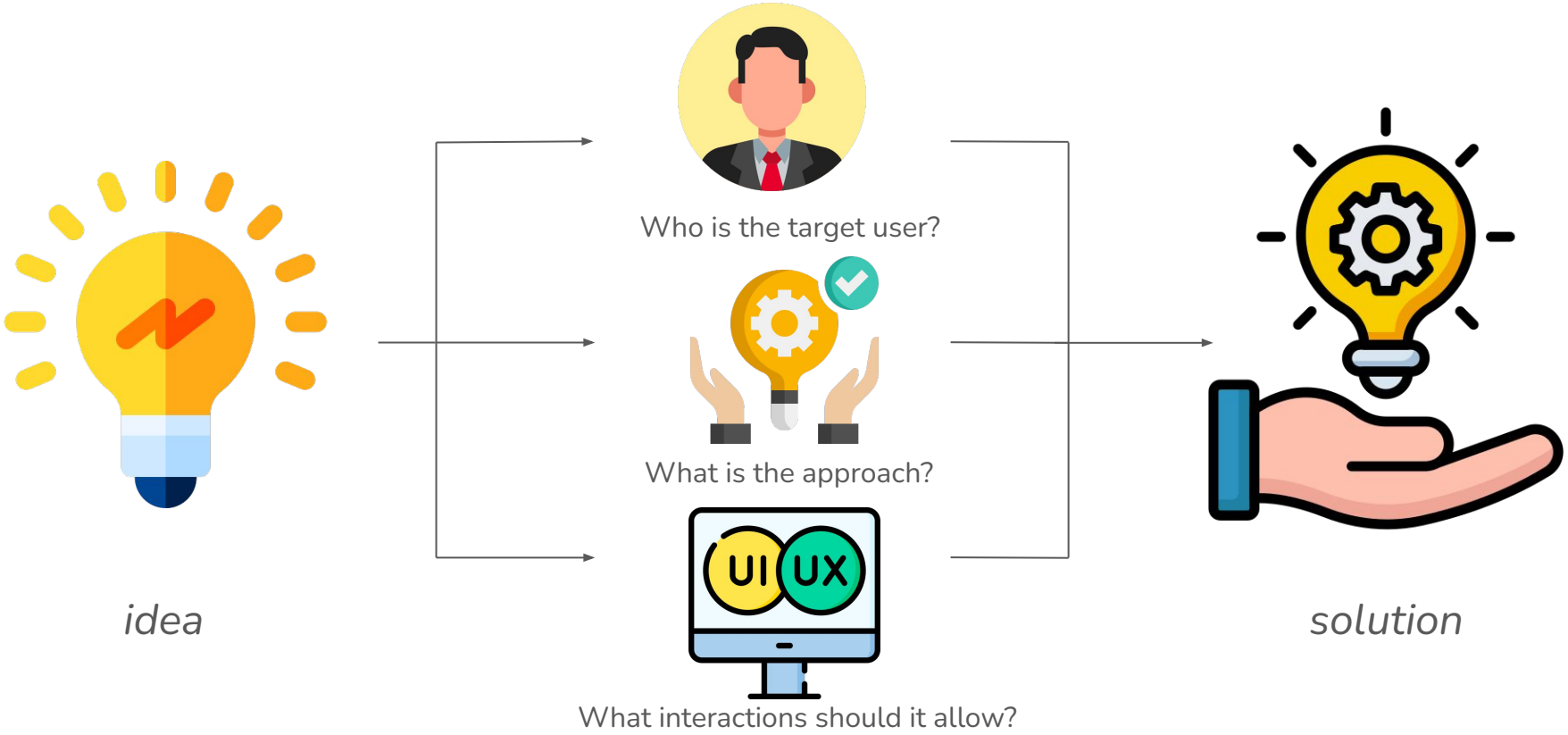
Who is the target user?



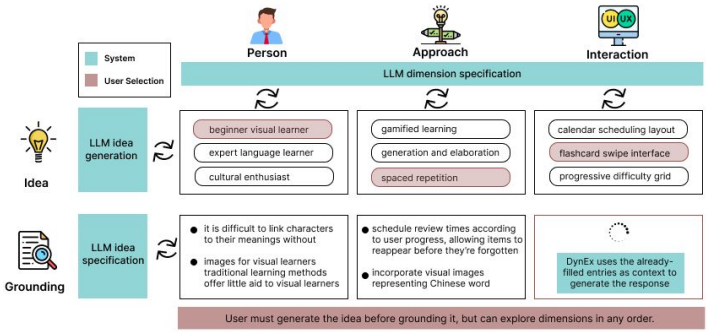
What is the approach?

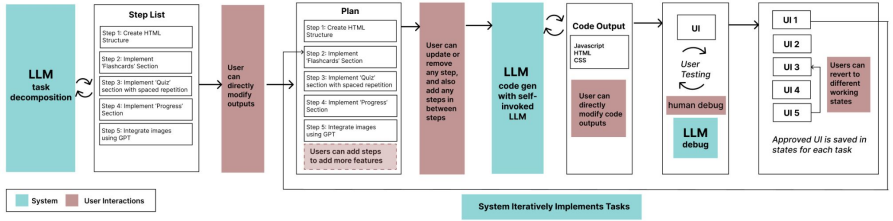
solution

We need to specific the problem! What is the **interaction paradigm**?

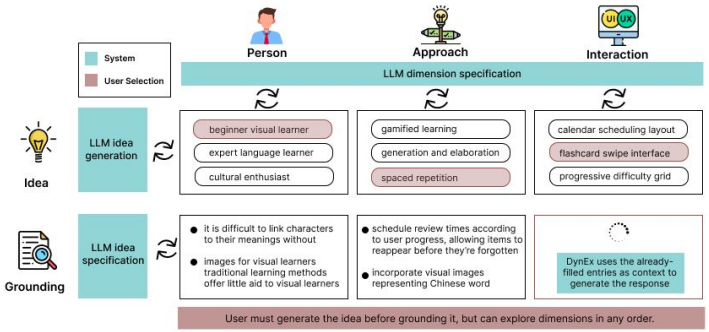


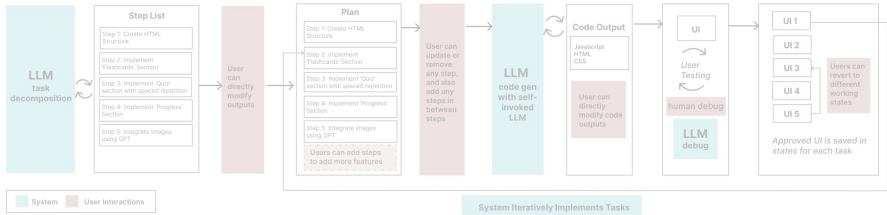
Design Exploration + Implementation





Design Exploration + Implementation





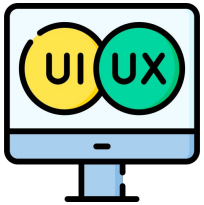
Design Matrix: Help users specify the user, approach, and interaction paradigm.

Dimensions

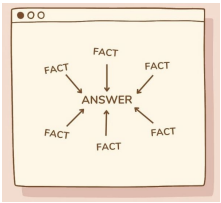
Person

Approach

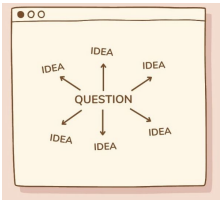
Interaction



Levels of specificity



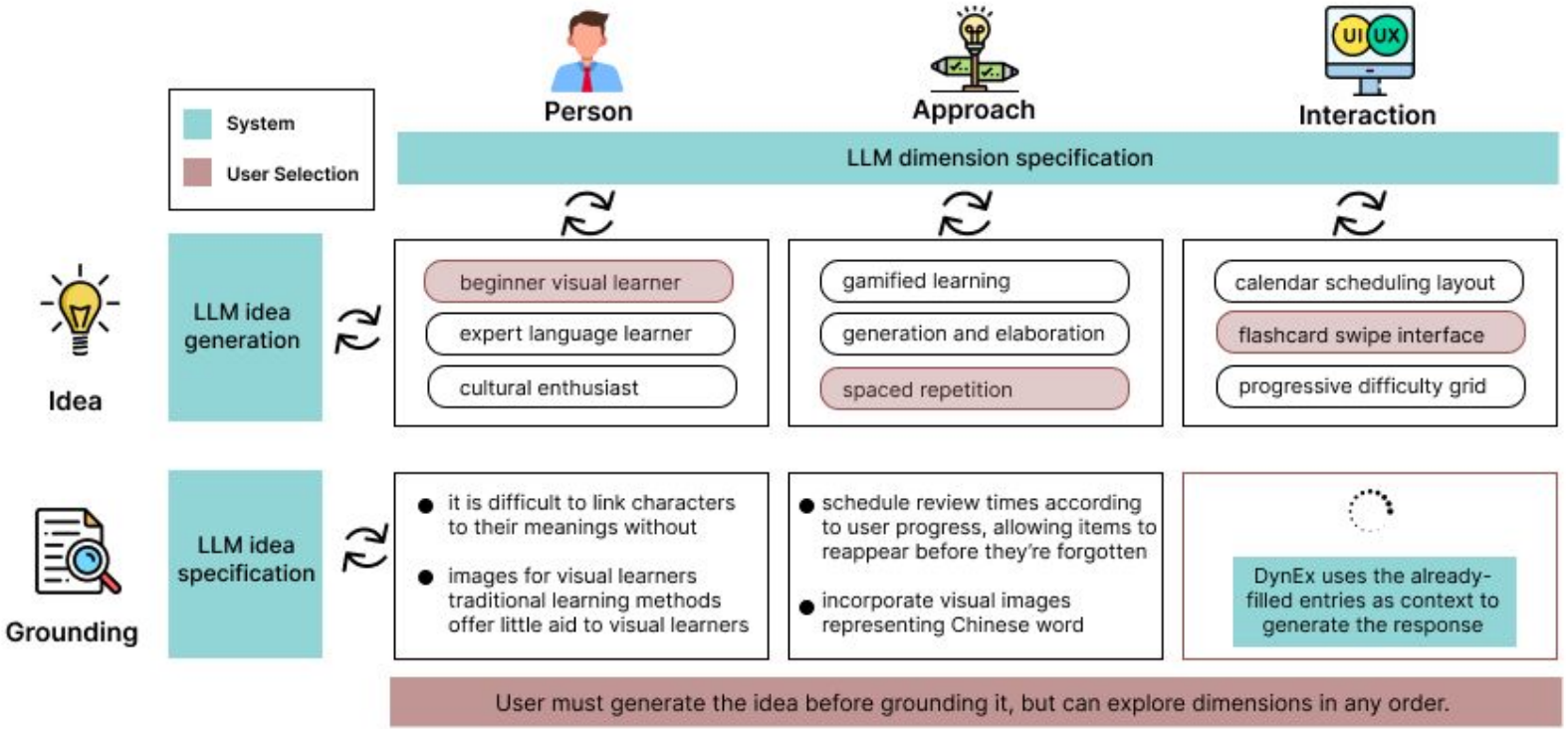
Idea



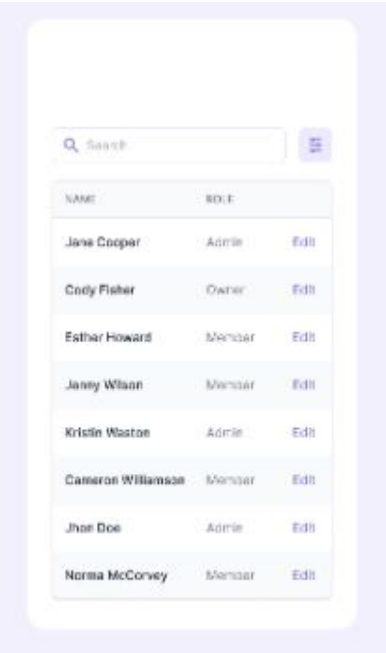
Grounding

Person:Idea	***Approach:Idea***	***Interaction:Idea***
Person:Grounding	***Approach:Grounding***	***Interaction:Grounding***

The Design Matrix uses Gen AI to suggest ideas and grounding with respect to the other ideas/groundings

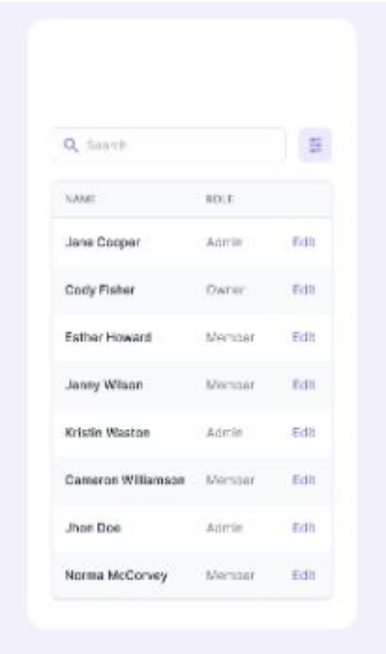


User Interaction Paradigms

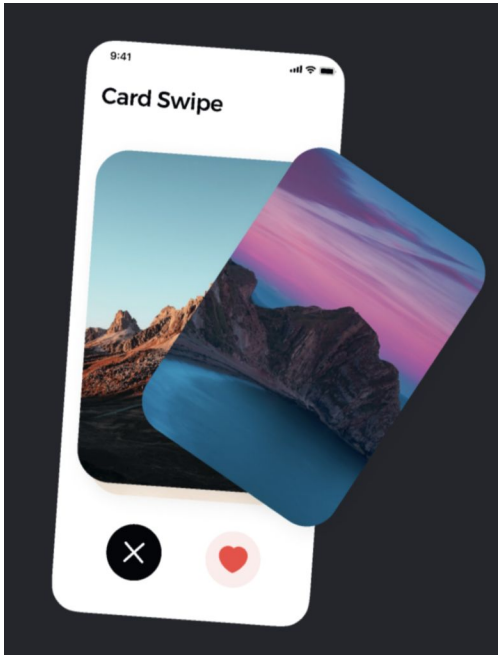


Table

User Interaction Paradigms

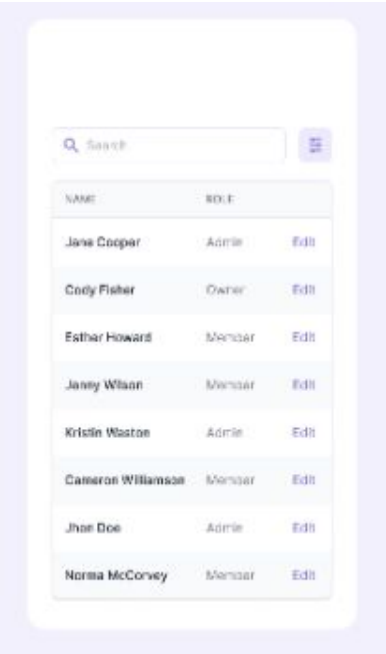


Table

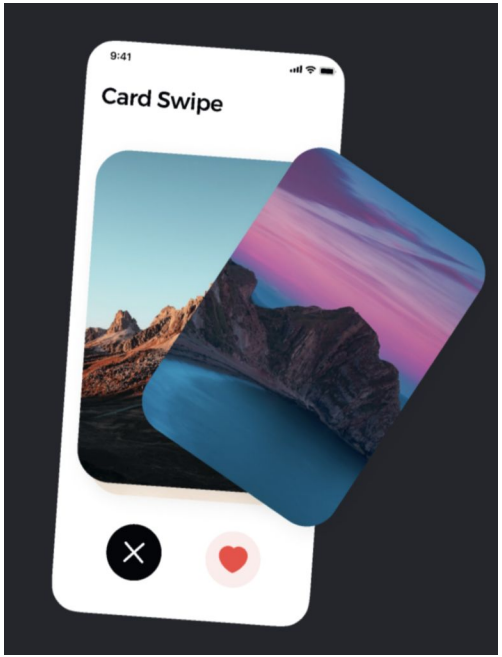


Card Swipe

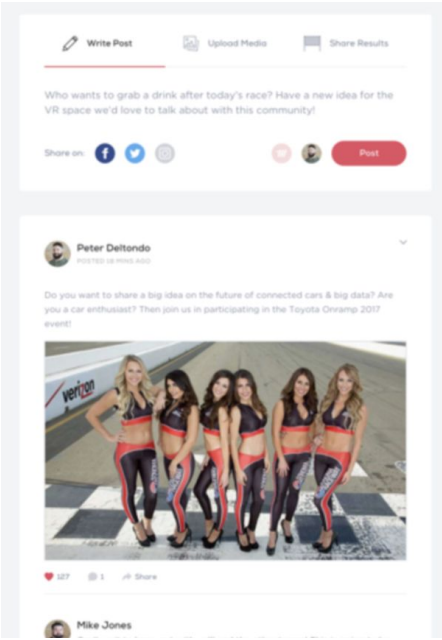
User Interaction Paradigms



Table

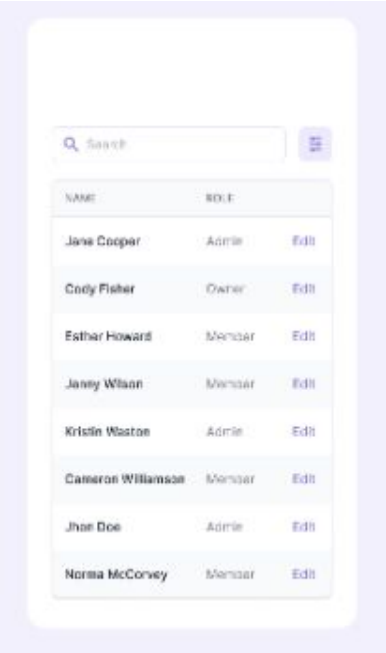


Card Swipe

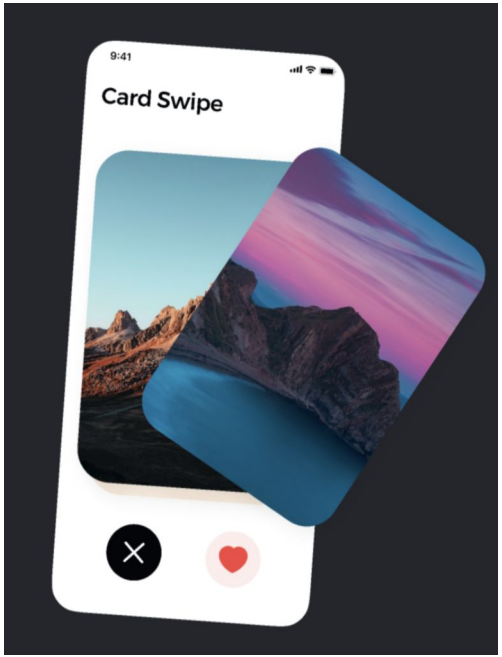


News Feed

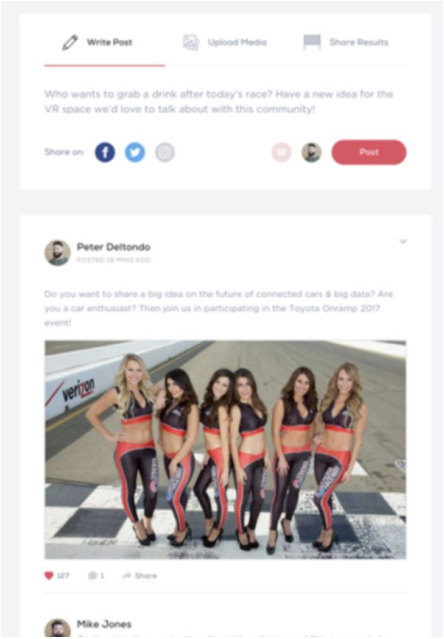
User Interaction Paradigms



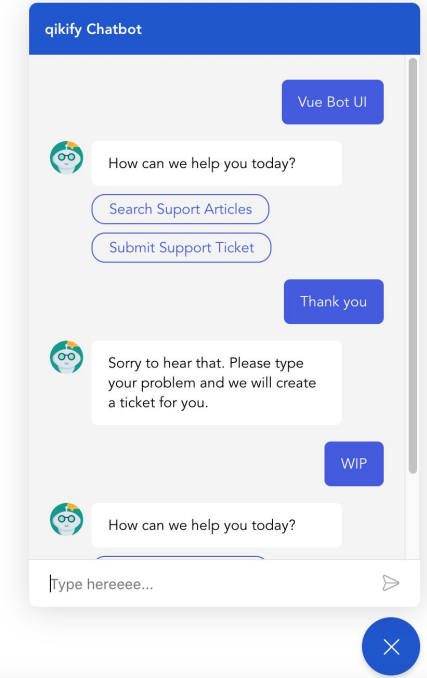
Table



Card Swipe

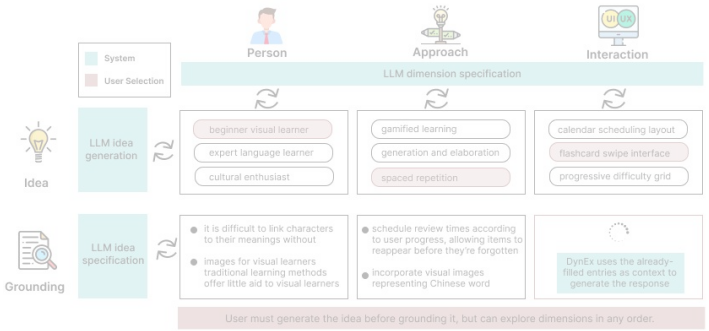


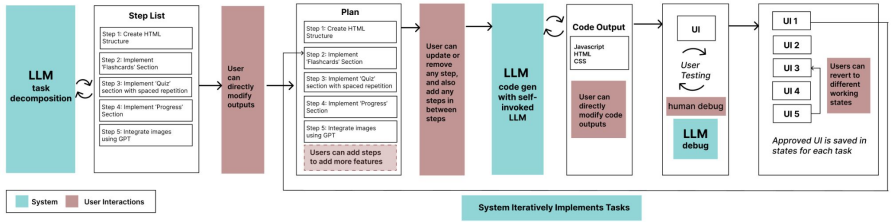
News Feed



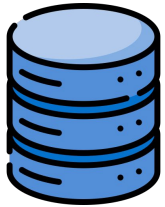
Chatbot

Design Exploration + Implementation





Design Requirement Agent



PROJECT REQUIREMENTS

BRAINSTORM

- [ ] GPT
- [x] Images
- [x] Faked Data
- [ ] ChartJS
- [ ] GoJS

UPDATE TOOLS REQUIREMENT

GENERATE NEW SPEC

Spec

Application Layout:
- Divide the interface into three main sections: "Flashcards," "Quiz," and "Progress."
- The "Flashcards" section displays the Chinese character/word along with its corresponding image for visual association.
- The "Quiz" section presents the character/word and image, prompting the user to input the meaning.
- The "Progress" section shows the user's performance metrics, such as accuracy rate and words mastered.

User Interactions:
- In the "Flashcards" section, users can click through the flashcards to study the character/word and its associated image.

UPDATE SPEC

REGENERATE PLACEHOLDER DATA

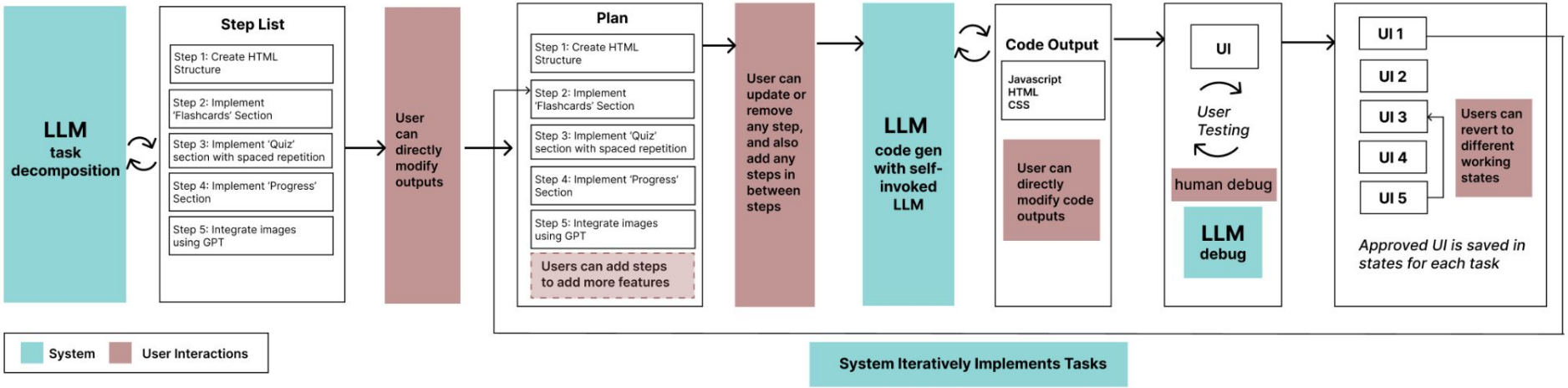
Data Input Suggestions

Data Input

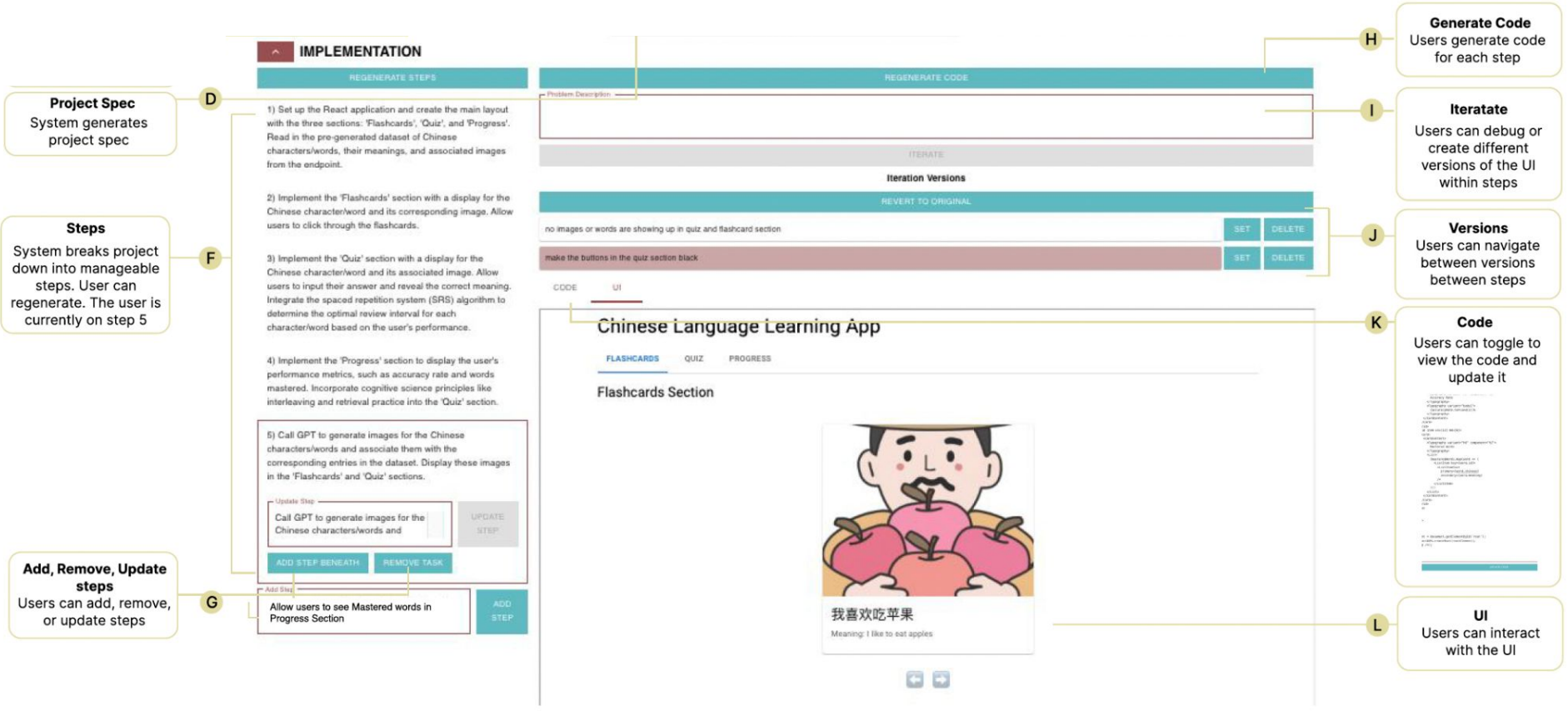
```
[
  {
    "id": 1,
    "chinese": "我学习中文",
    "meaning": "I study Chinese",
    "imagePath": "images/studying.jpg"
  },
  {
    "id": 2,
    "chinese": "你好吗?",
```

UPDATE FAKED DATA

Dynamic Iterative Implementation Agents

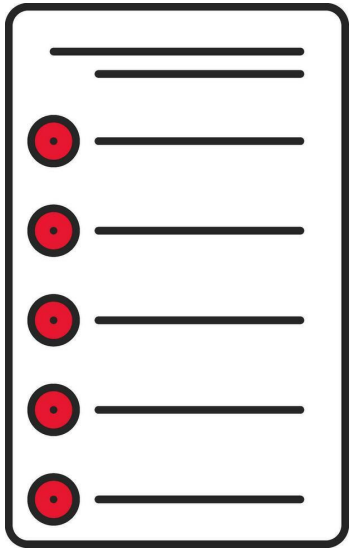


Iterative Development with Step-by-Step Implementation



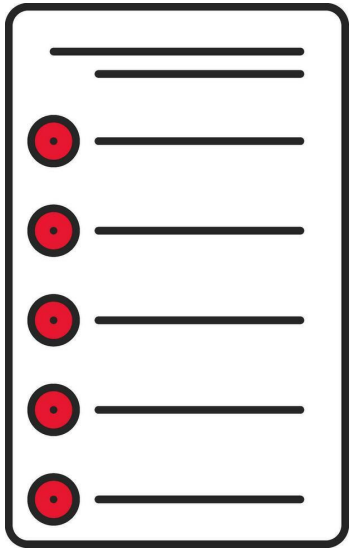
Modular Step-by-step Implementation

System breaks down implementation plan into **steps** and generates code for each step

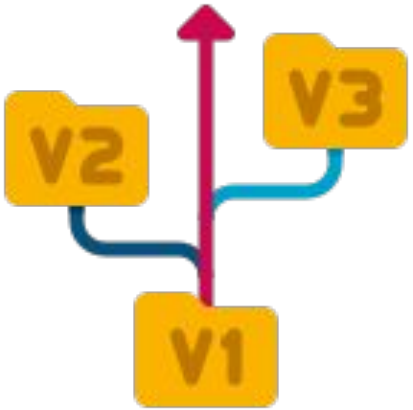


Modular Step-by-step Implementation

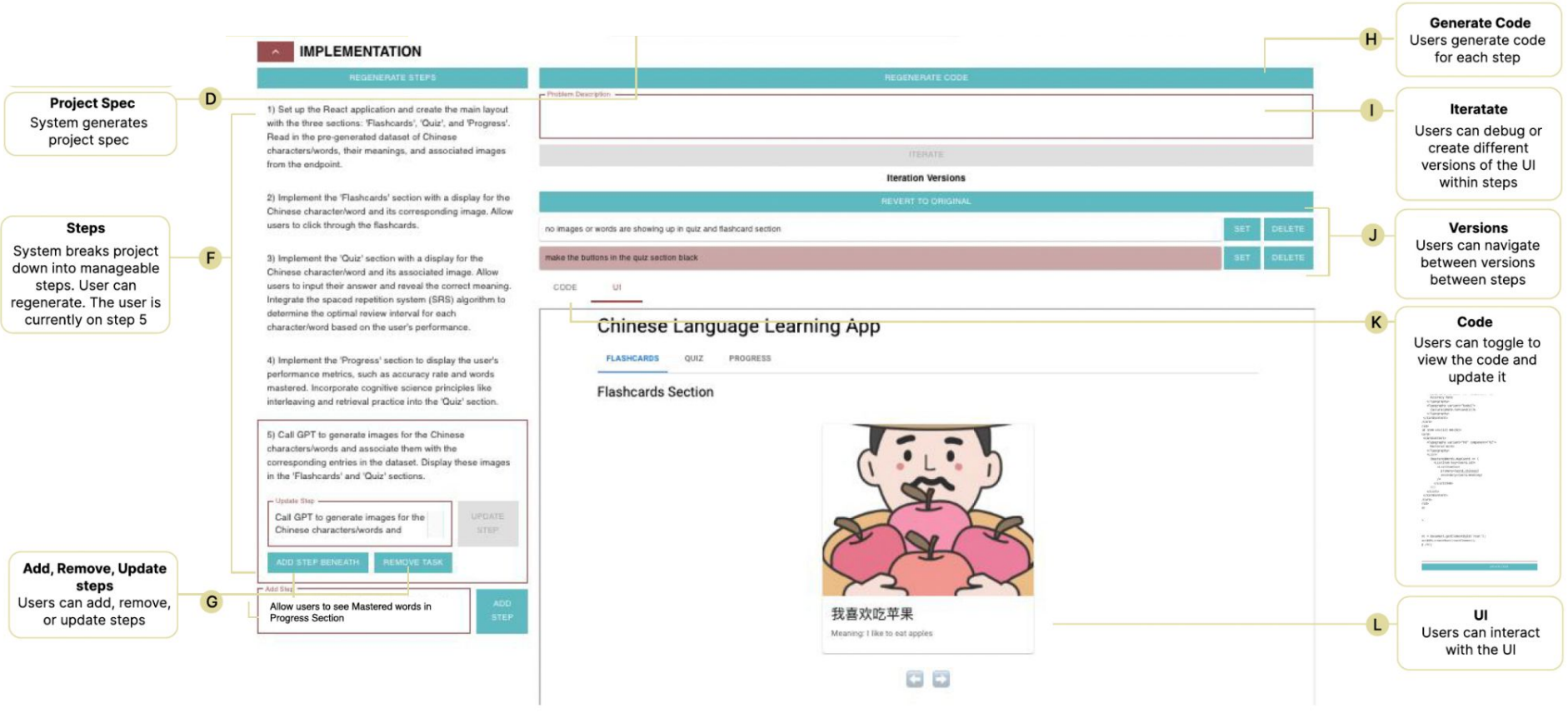
System breaks down implementation plan into **steps** and generates code for each step



Users can **add**, **remove**, and **update** steps. Steps provide a natural form of **version control**.



Self-Invoking LLMs

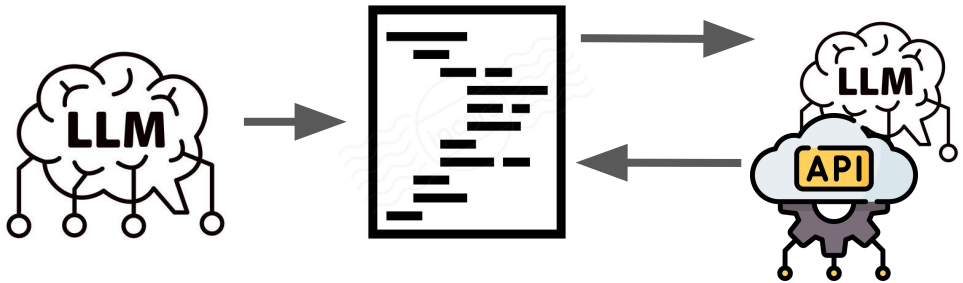


Self-Invoking Multi-Modal LLMs

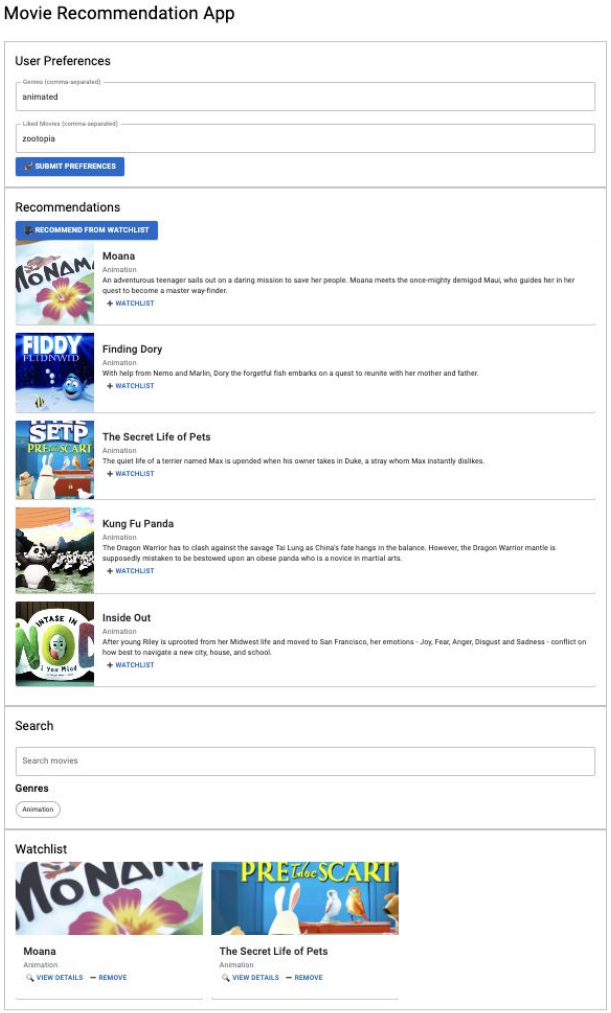
Generative AI APIs are **powerful** and **versatile**, and excel at handling a wide variety of tasks.

Our generated code can call GPT to generate **dynamic data** and **create images**.

We can create more **realistic** applications that can more accurately mimic a user's experience



Examples



Movie recommendations (and images) are made by calling GPT within the app to generate the related movies dynamically

Book Recommendation App

Recommended Books

To Kill a Mockingbird

Harper Lee

A classic story of racial injustice and the destruction of childhood innocence in the American South, told through the eyes of a young girl.

1984

George Orwell

A dystopian novel set in a totalitarian future society where the protagonist wrestles with oppression and state-imposed surveillance.

Search for Books

Search

SEARCH

Favorite Books

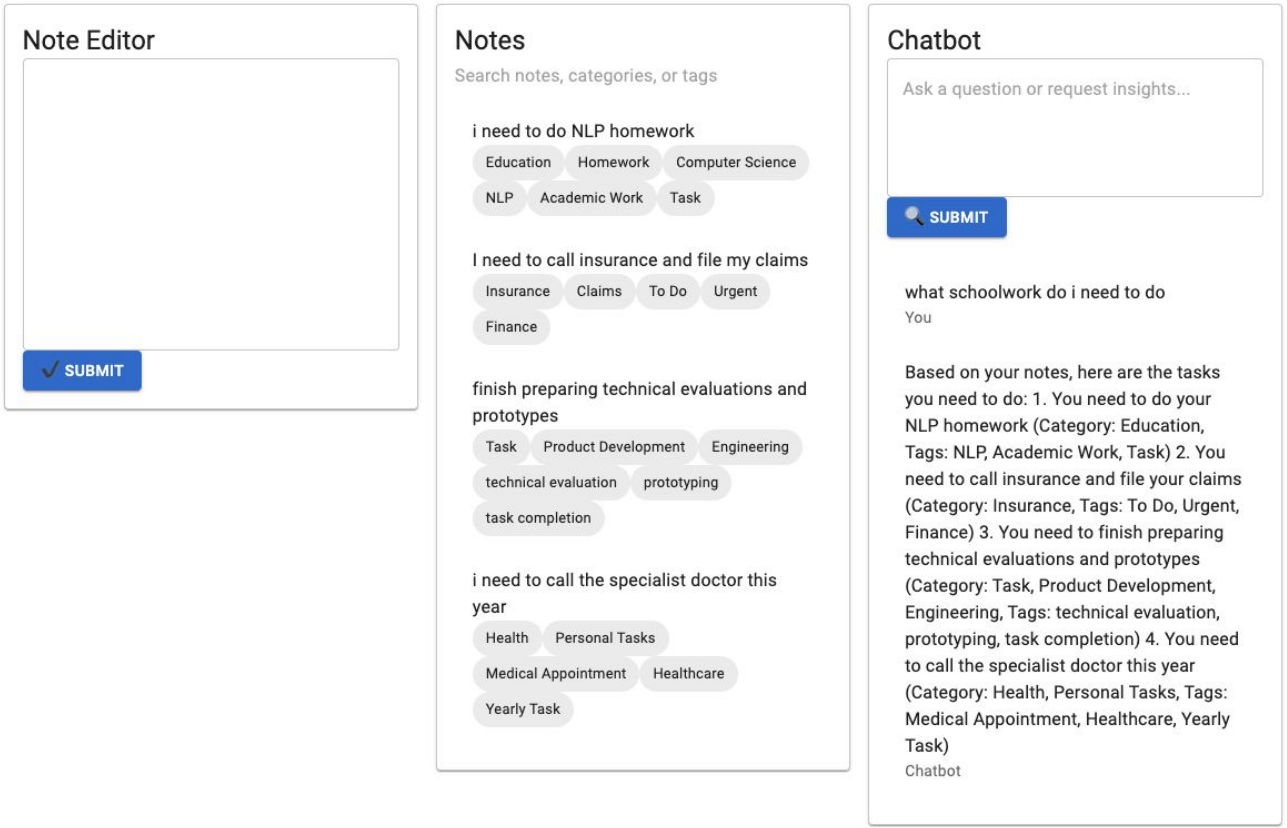
The Hunger Games by Suzanne Collins

Harry Potter and the Sorcerer's Stone by J.K. Rowling

The Book Thief by Markus Zusak

Book recommendations, images, and summaries are made by calling GPT within the app dynamically

Note-Taking and Sensemaking Application



Clustering of notes is done dynamically with GPT

User Evaluation Research Questions

RQ1: *[Divergence]* To what extent does DynEx enable divergent exploration within a problem space?

RQ2: *[Convergence]* To what extent does DynEx allow users to better develop their ideas?

RQ3: *[Implementation]* To what extent does DynEx enable the code to realize a complex idea?

RQ4: *[Overall]* To what extent does DynEx allow for a better prototyping experience?

Claude

Book Recommendations UI

Initial UI Prompt
Users prompt Claude with their initial idea. Claude creates a UI

A create a ui for book recommendations

Certainly! I'll create a React component for a book recommendations UI. This component will display a list of recommended books along with their details and a simple rating system.

Book Recommendations UI
Click to open component

I've created a React component for a book recommendations UI. Here's a breakdown of the component:

1. The `BookRecommendation` component displays individual book recommendations, including:
   - Title and author
   - Description
   - Star rating (using Lucide icons)
   - A like button that toggles when clicked
2. The `BookRecommendations` component renders a list of book recommendations using sample data.
3. The UI uses components from the shadcn/ui library, such as Card and Button, for a polished look.
4. Tailwind CSS classes are used for styling and layout.
5. The component is responsive and will adjust to different screen sizes.

To use this component, you would need to have the shadcn/ui components installed in your project. You can customize the book recommendations by modifying the `recommendations` array in the `BookRecommendations` component.

Would you like me to explain any part of the code in more detail or make any modifications to the UI?

Iterate
Users iterate and add features by further chatting and adding prompts.

B i want a section to save my liked books

Certainly! I'll modify the component to include a section for saved (liked) books. We'll add functionality to move liked books to a separate "My Liked Books" section.

Book Recommendations UI with Liked Books Section
Click to open component

I've updated the Book Recommendations UI to include a section for liked books. Here

Chat
Users prototype through chat

C Reply to Claude...

Claude 3.5 Sonnet

Book Recommendations UI | Preview | Code

D

Book Recommendations

Search books or | All

Recommendations

The Great Gatsby
F. Scott Fitzgerald

A classic novel about the American Dream in the Roaring Twenties.

Genre: Classic

Like

To Kill a Mockingbird
Harper Lee

A powerful story of racial injustice and loss of innocence in the American South.

Genre: Fiction

Like

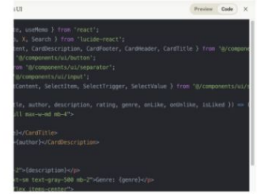
1984
George Orwell

A dystopian novel set in a totalitarian society, exploring themes of surveillance and control.

Genre: Science Fiction

Like

Artifact
Users can Interact with the Preview and view the code. However, they cannot edit the code.



Versions
Users can view previous verisons

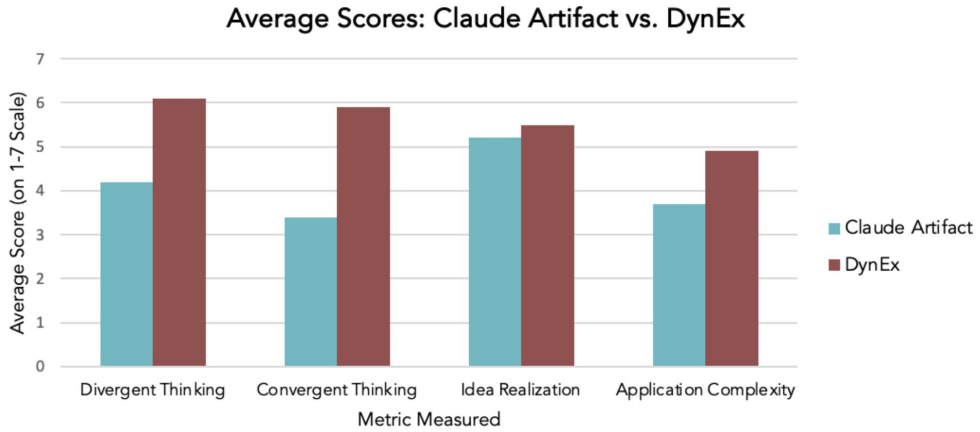
E

Version 3 of 3

Publish

Results

- DynEx **inspired new solutions**, allowed users to **explore a problem space**, and **further developed users ideas** *(p=0.05 level)*.
- DynEx enables users to create more **complex**, **feature-rich**, and **intuitive** applications that emulate a true user experience *(p=0.05 level)*



P6 - Friend-Activity-Joining Application

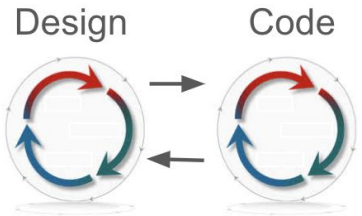
"**[DynEx] suggested many features**… such as **how to match people with friends**, **how to visualize the friend-sharing [component]**, **how to [join a friend's] friend experiences**, **how to share experiences with friends**.. it created a pretty robust social ecosystem surrounding the calendar experience."

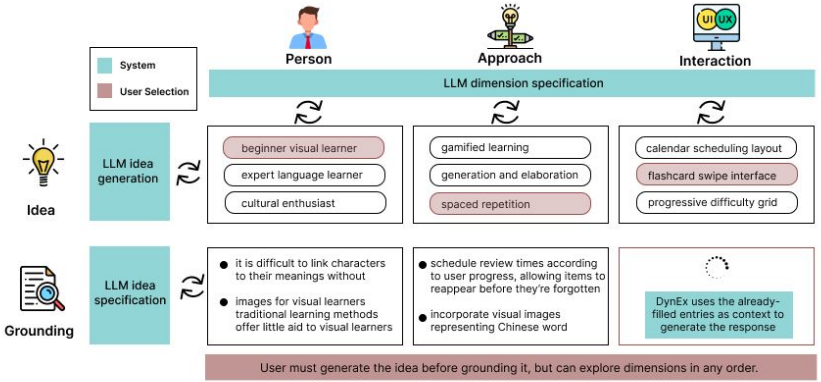
"[DynEx] incorporated really different, very distinct UI features that weren't very connected to each other at [face value] really well"

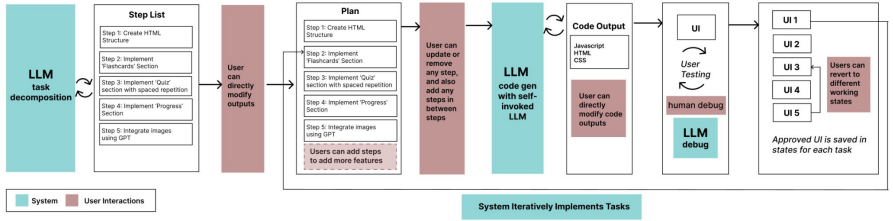
P8 - Concert Ticket Aggregator

“It was **intuitive**. It was **feature-rich**. It had all the important features, like **sorting by columns** and **sorting by genre, value for money,** etc… I liked the **conciseness** of the information… contrary to [Claude Artifact’s] prototype which was… not-fit for this use case”

Dynex: Bridging Design and Code







Iterative Problem Specification (Design)

Modular Code Generation and Testing

Bridging Design and Code with Gen AI

Dynex

- Iterative Problem Specification
- Iterative Development

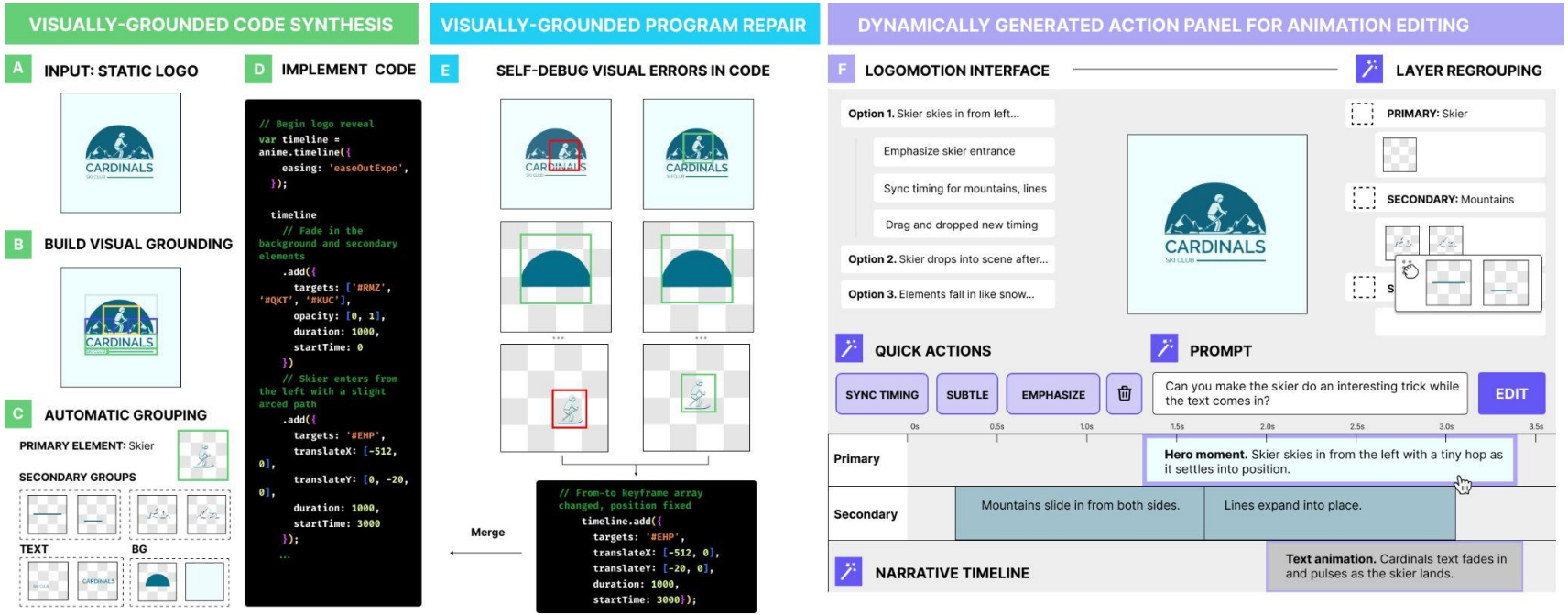
LogoMotion

- Self-debugging loop
- Iteration with AI Editing widgets

Double Agents

- Testing with user simulation
- Iterating on designs with AI suggestions

LogoMotion: Visually Grounded Code Generation and Repair

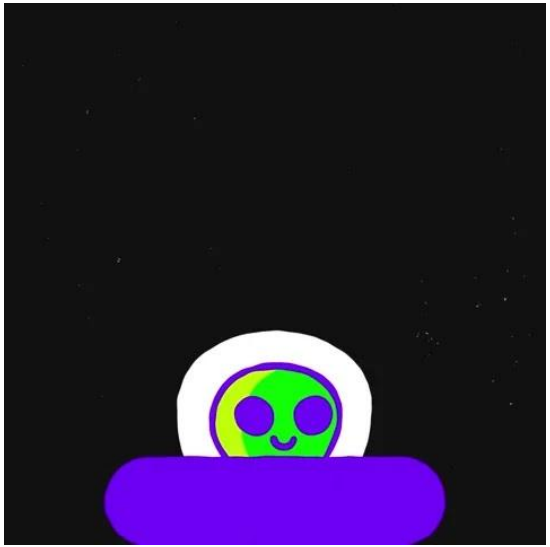


Vivian Liu, Rubaiat Habib Kazi, Li-Yi Wei, Matthew Fisher, Timothy Langlois, Seth Walker, Lydia Chilton.
LogoMotion: Visually Grounded Code Generation for Content-Aware Animation. CHI 2025.

Creating semantically meaningful motion is hard.

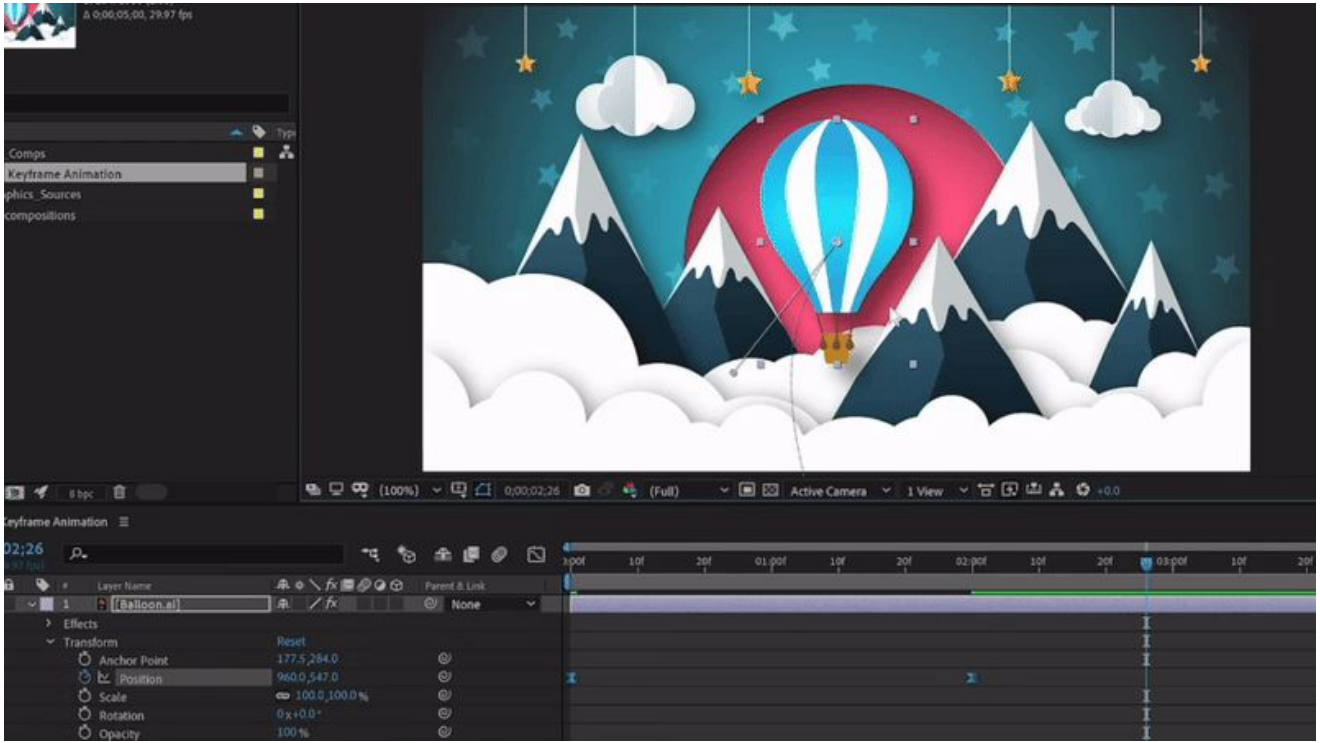
Cars should drive

Fans should spin

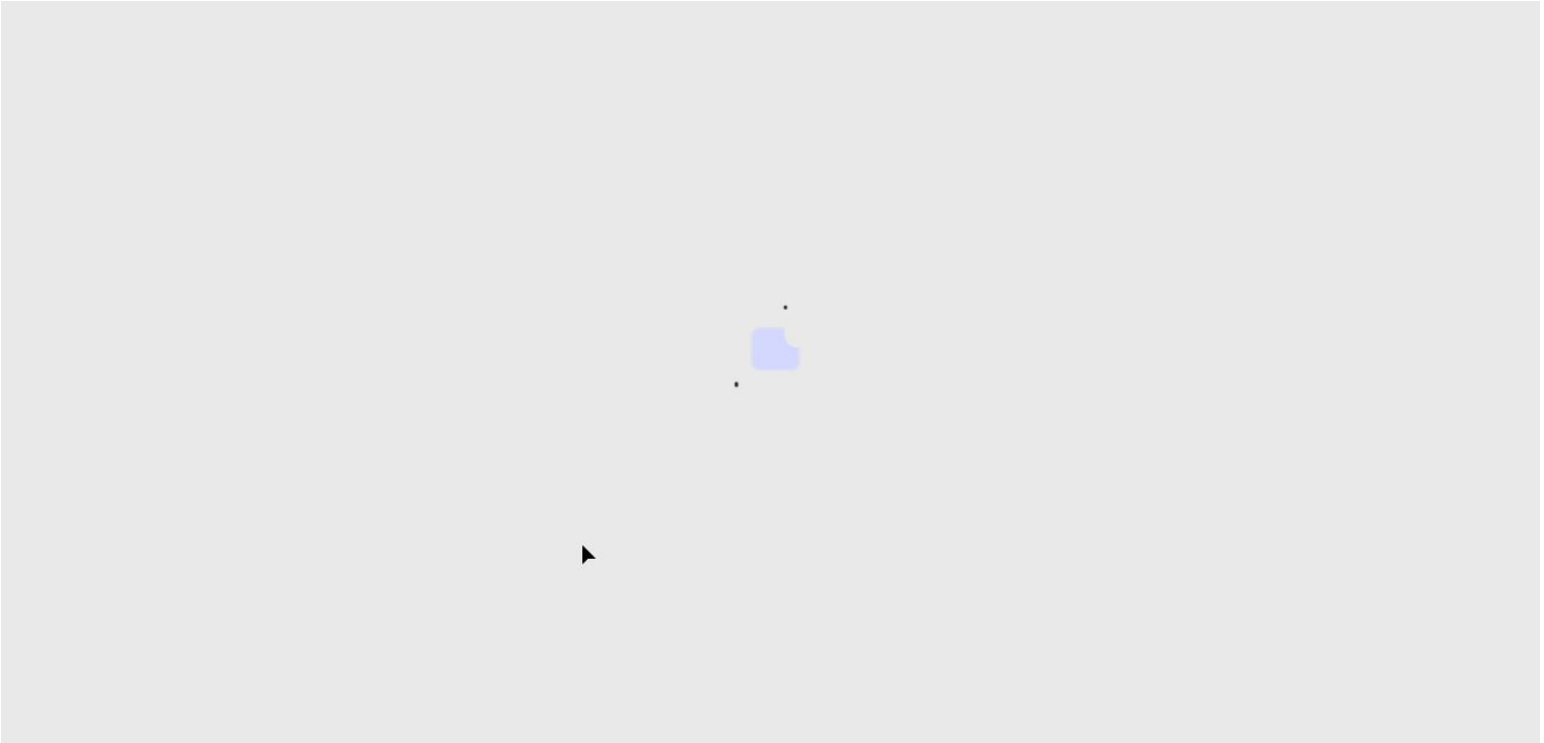


UFOs should take off

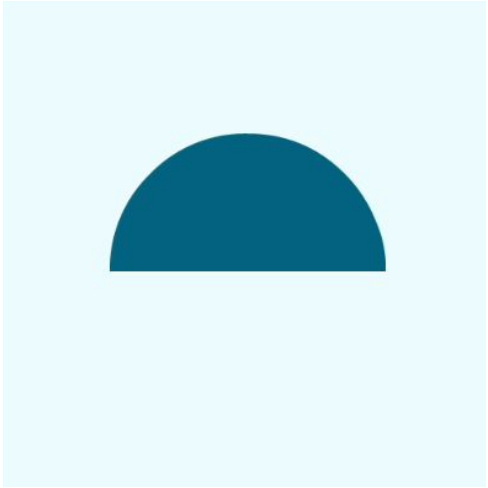
Animation UIs are complex



Templates are too rigid



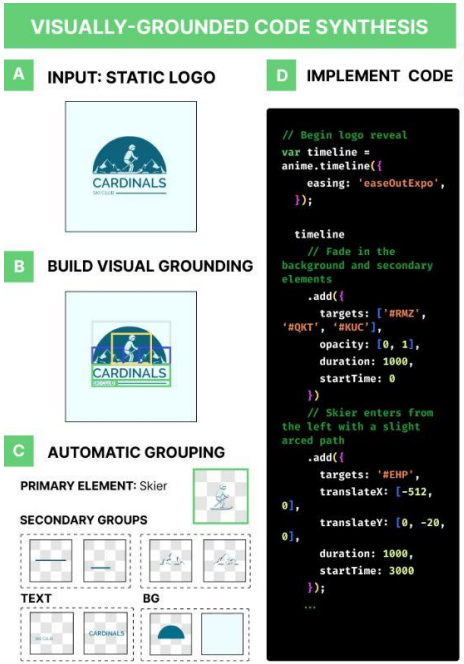
Human-agent interaction for animation gives users control and freedom



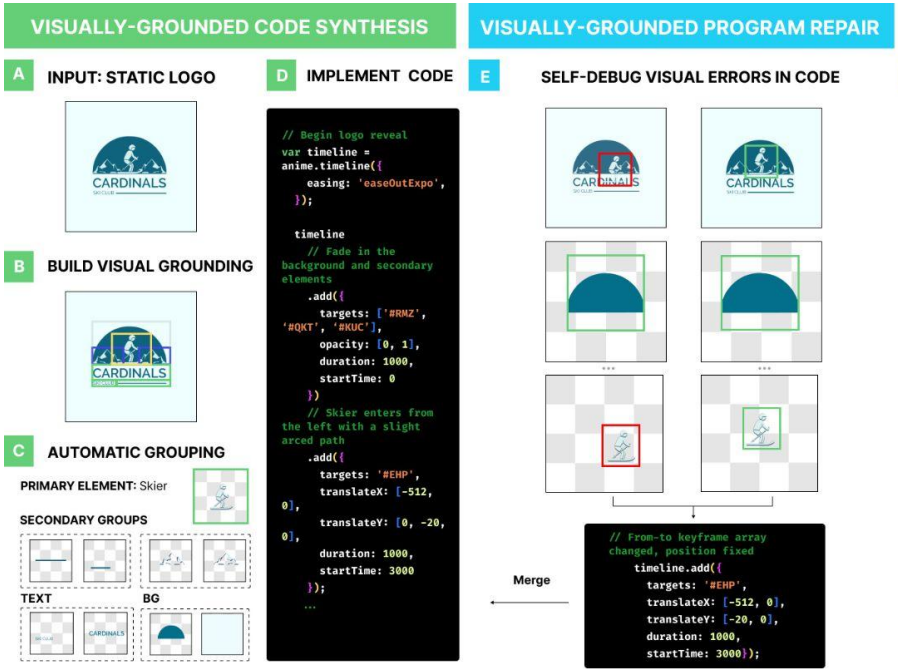
INPUT
PDF

OUTPUT
HTML PAGE + ANIMATION CODE

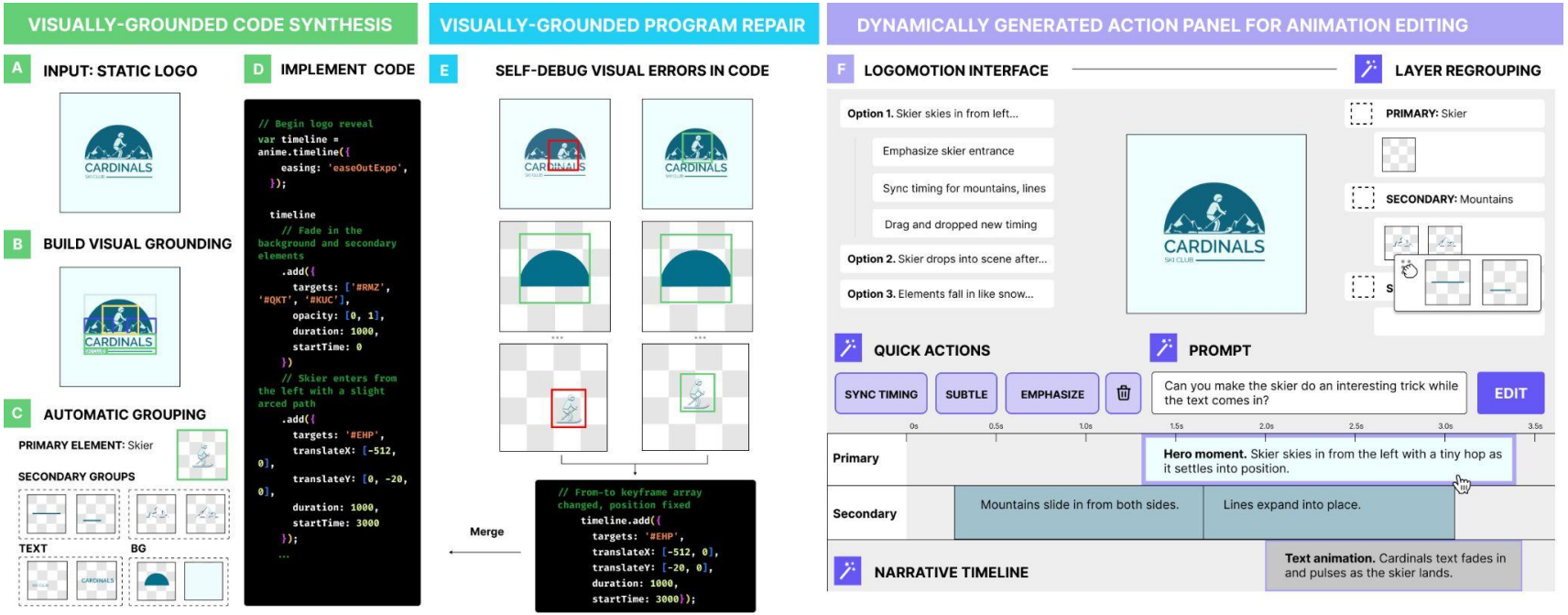
Agents can automatically author an animation



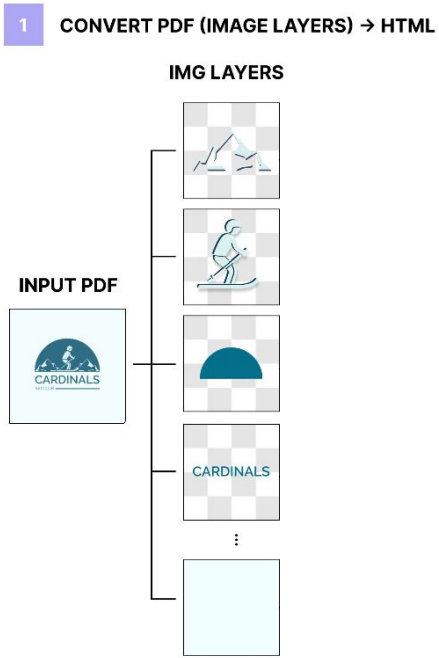
Agents can automatically debug and repair animations



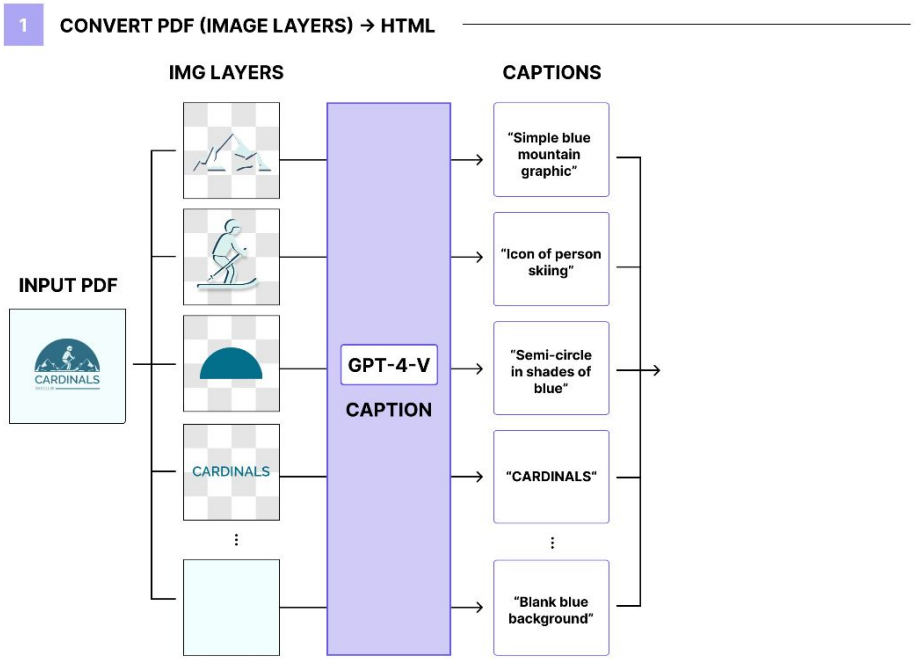
Agents Generate a UI for Editing the Animation



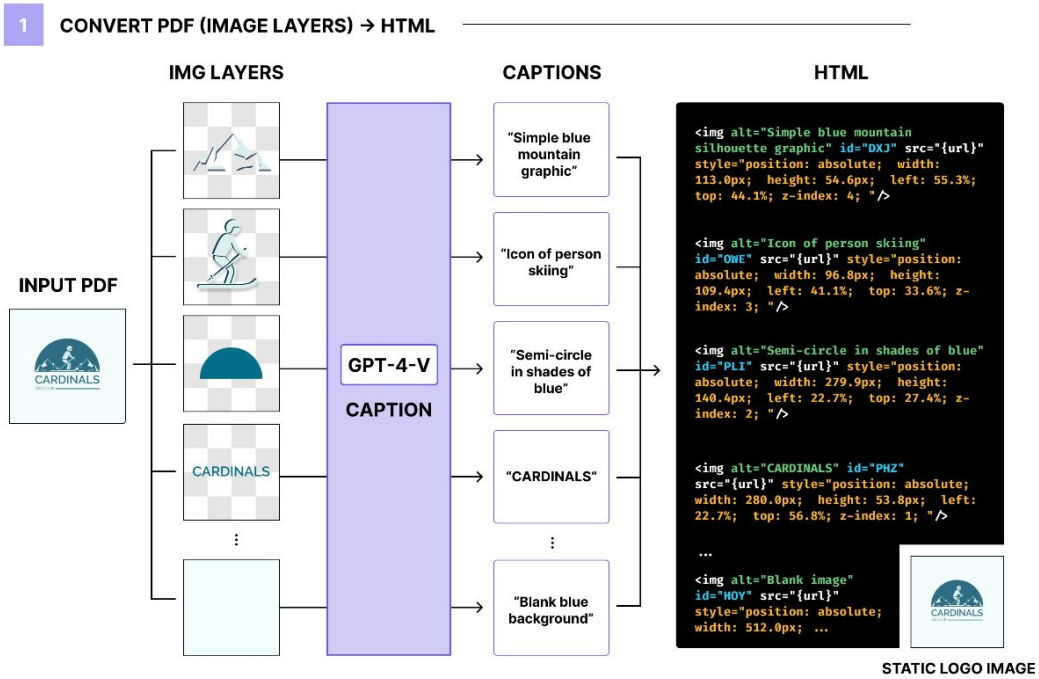
Agents visual understand the input.



Agents label the parts of the input

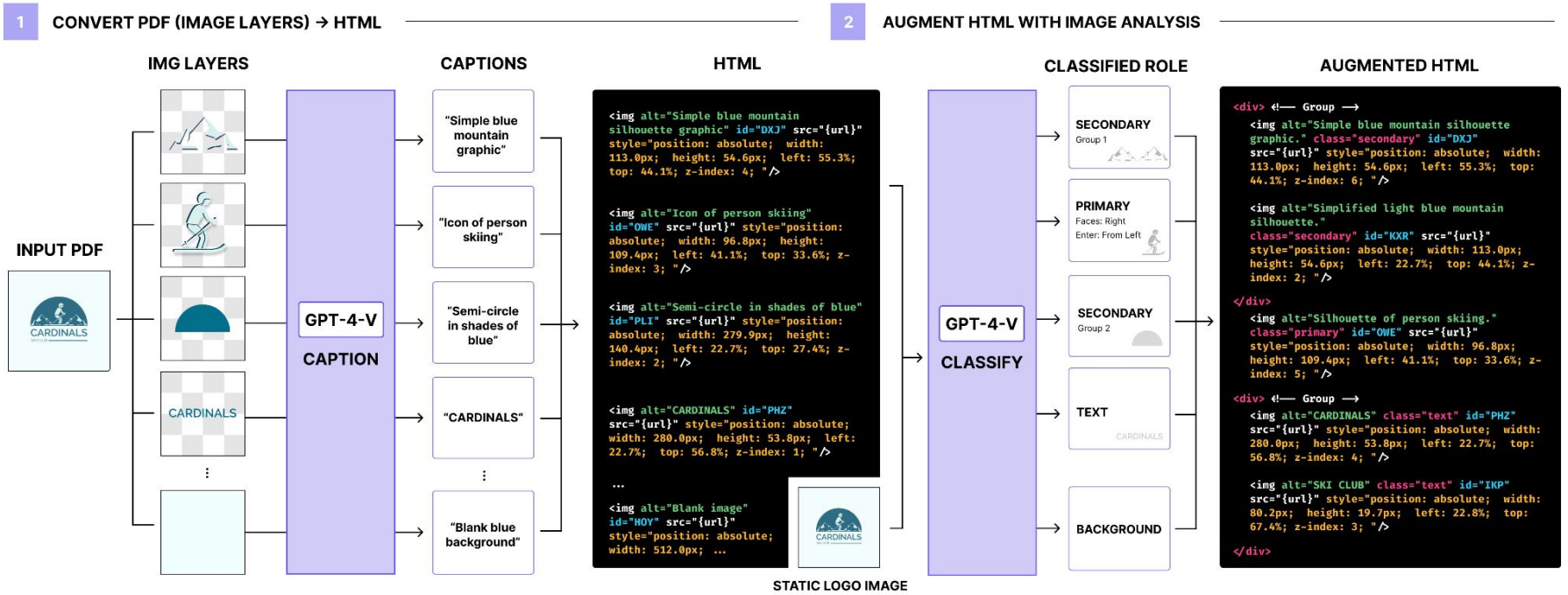


Agents represent the input as HTML

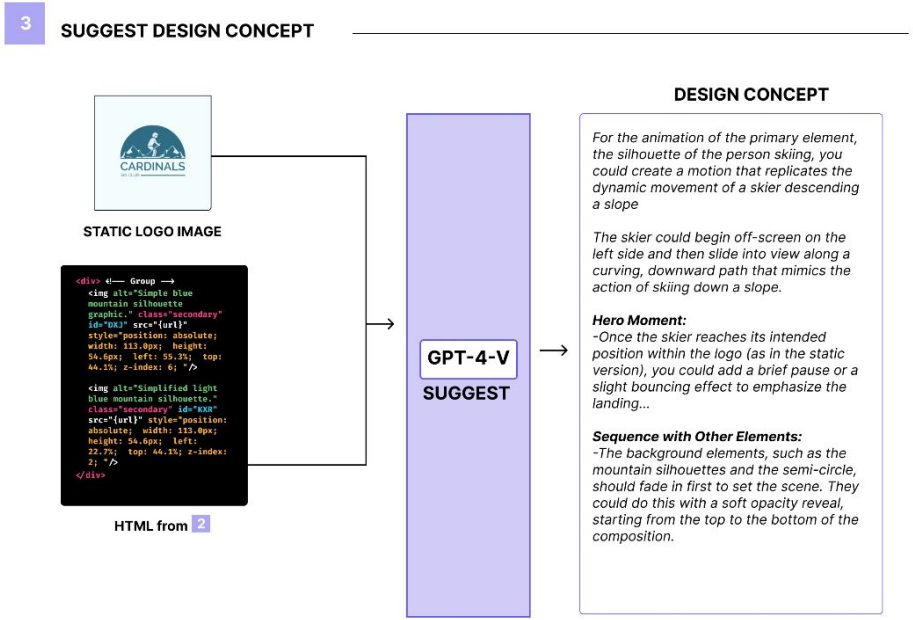


Augment the HTML with visual analysis

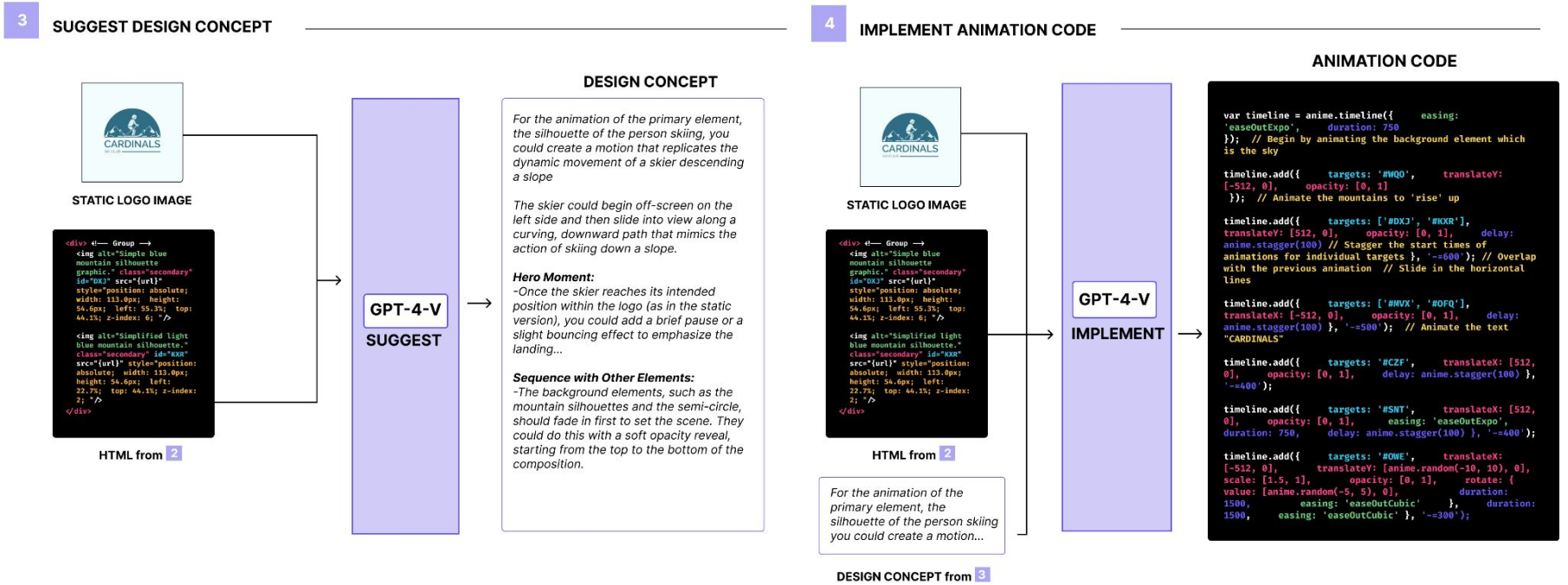
(Primary, secondary visuals, text, background, etc)



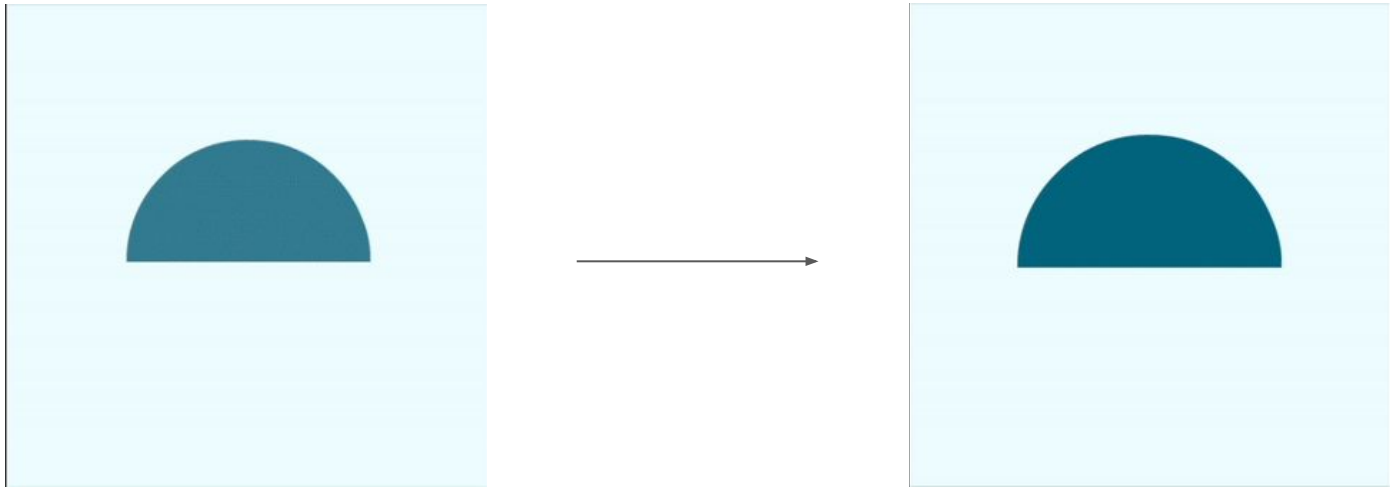
Agents suggest a design concept.



Agents implements animation code.



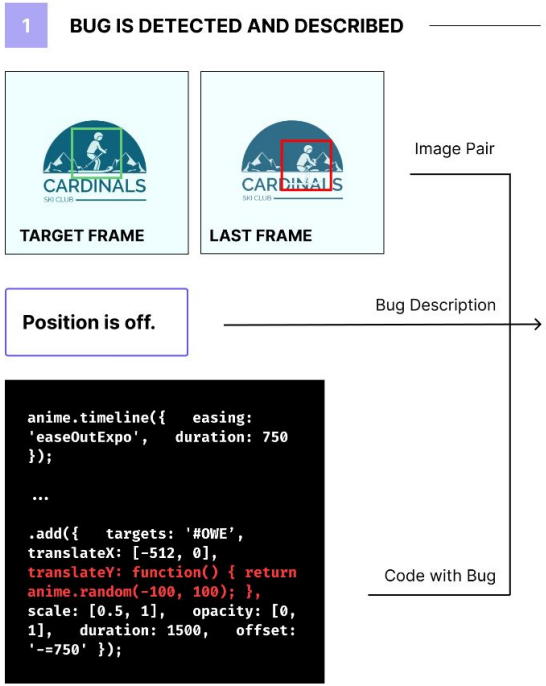
Agents can also detect a fix animation errors



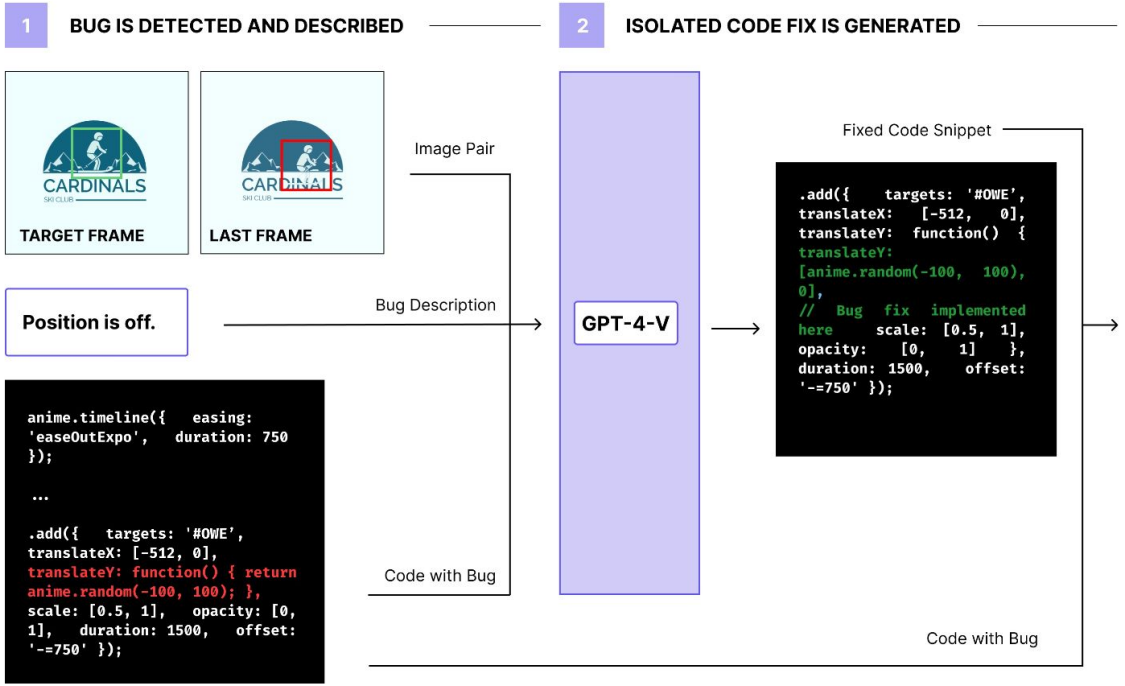
OUTPUT WITH ERROR

SELF-DEBUGGED OUTPUT

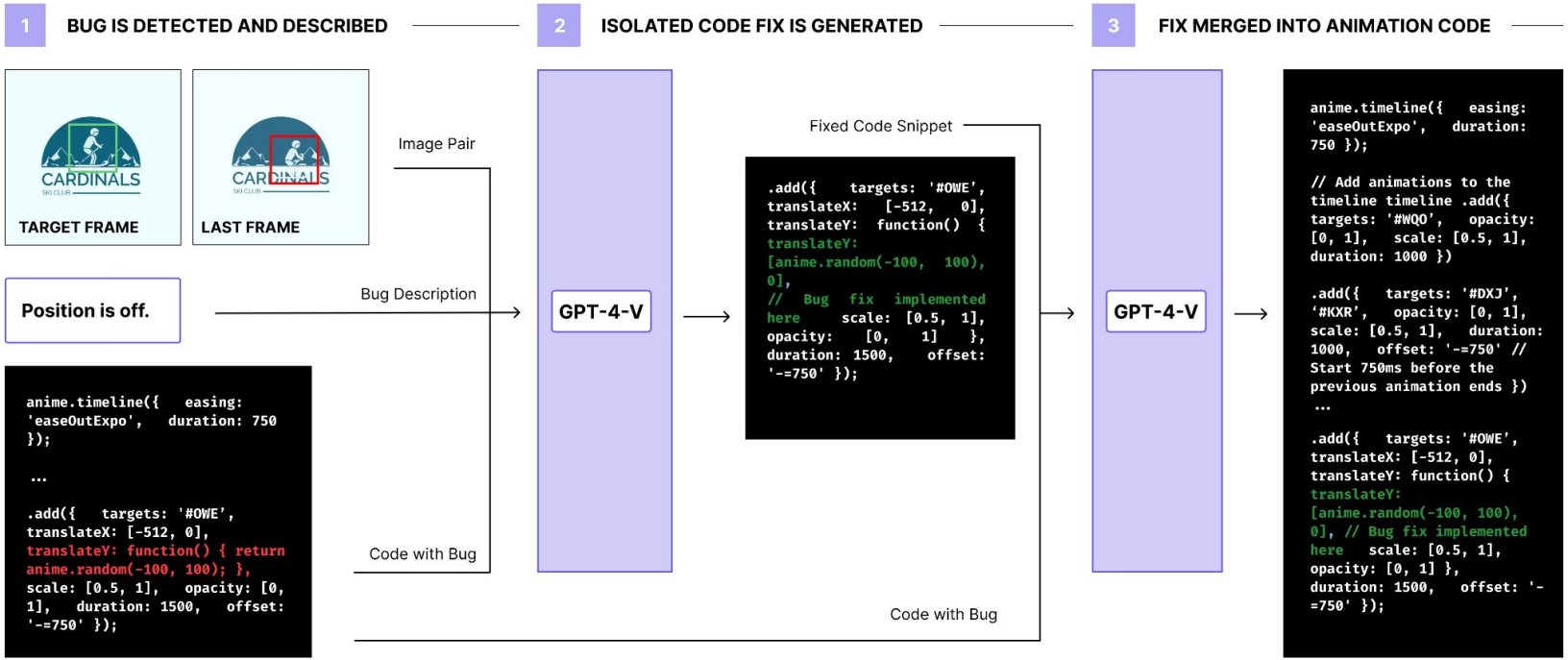
Self-debugging agents check for errors.



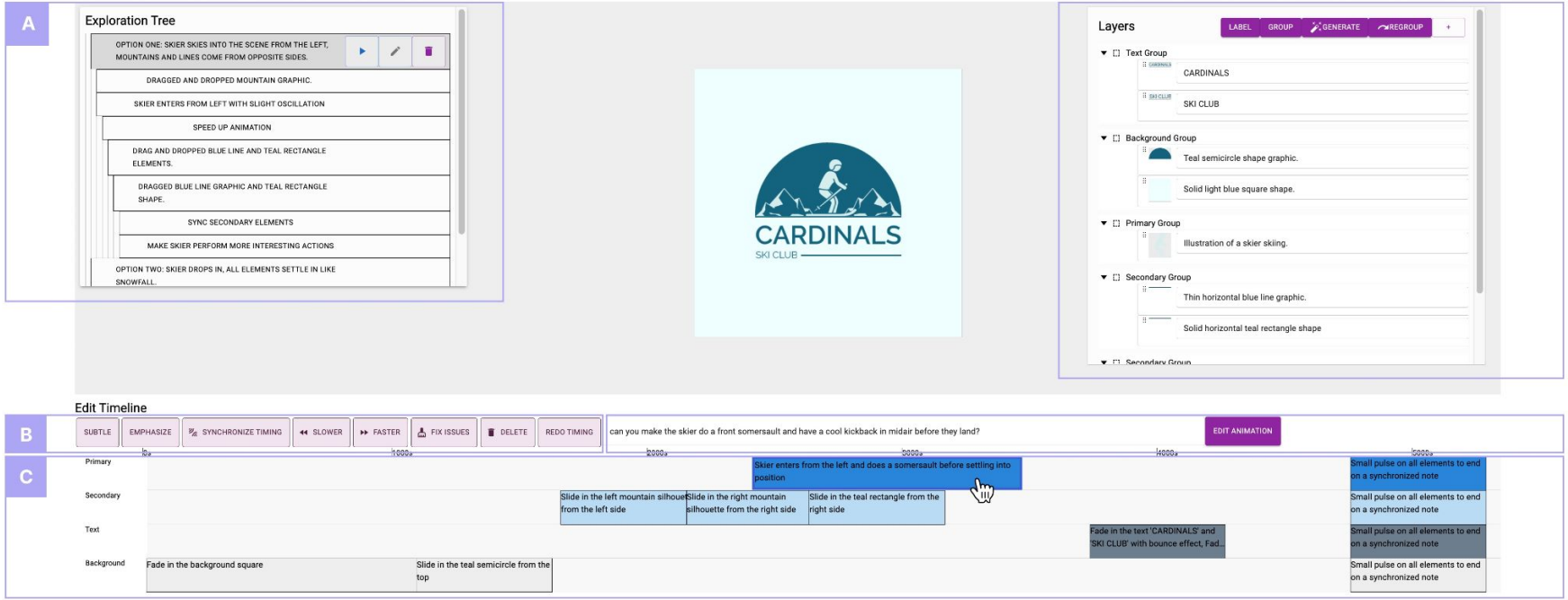
Self-debugging agents check for errors.



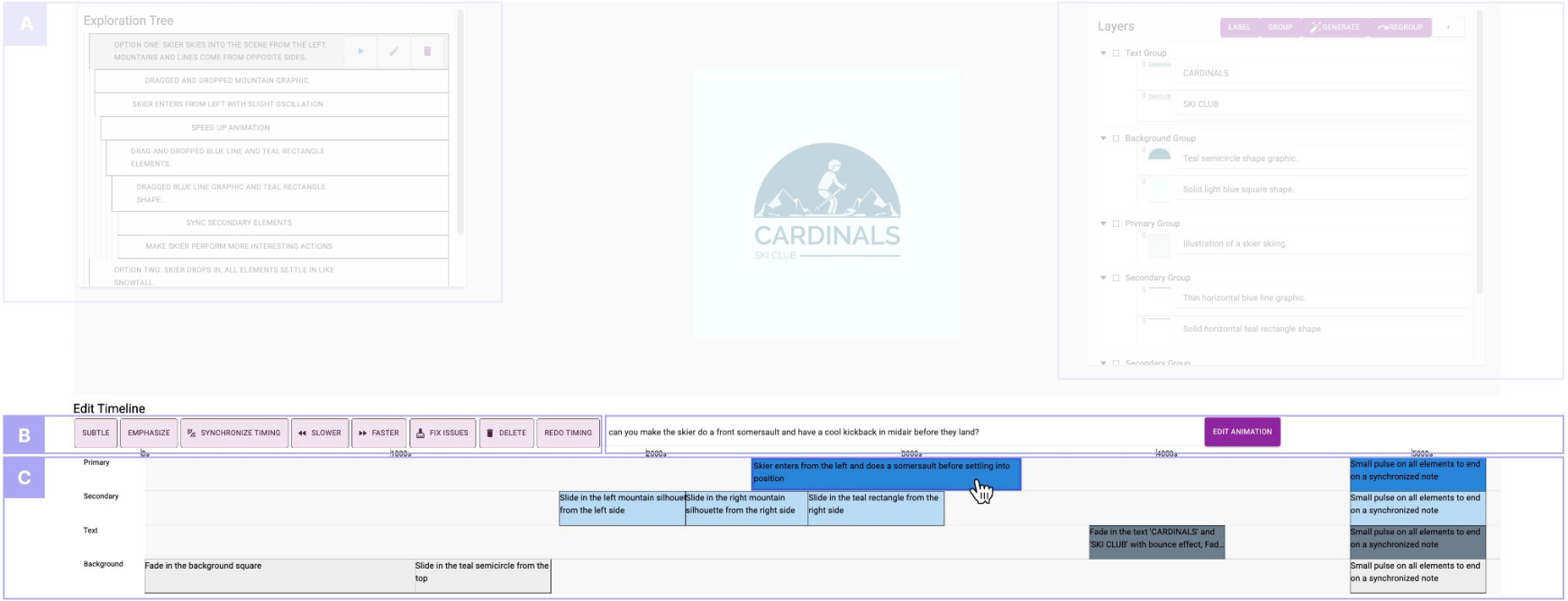
Self-debugging agents check for errors.



Agents activate an editing UI for users to improve animations or explore new ones.



A timeline widget can reorder animation blocks and adjust timing



Options

OPTION ONE

OPTION TWO

OPTION THREE

OPTION FOUR

Layers

LABEL GROUP GENERATE REGROUP +

Text Group

CARDINALS

SKI CLUB

Background Group

Teal semicircle shape graphic.

Solid light blue square shape.

Primary Group

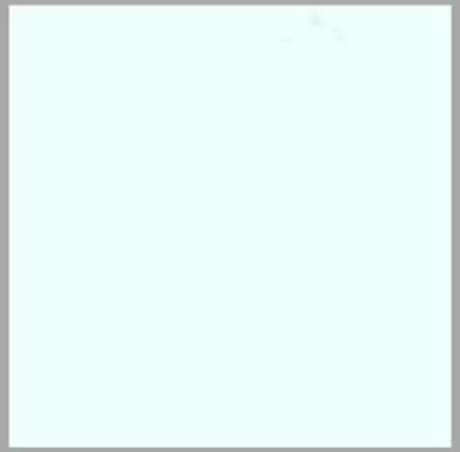
Illustration of a skier skiing.

Secondary Group

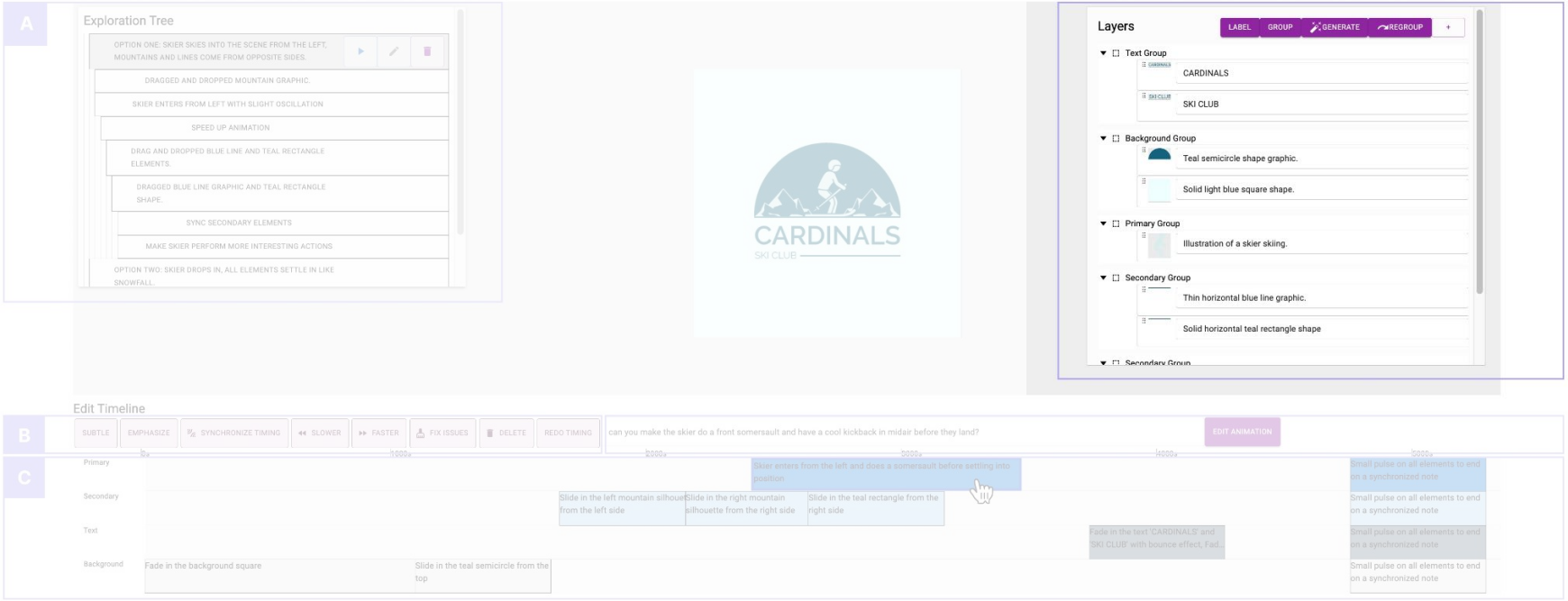
Thin horizontal blue line graphic.

Solid horizontal teal rectangle shape

Animation First Draft



A grouping widget can synchronize motion and timing of elements



Options

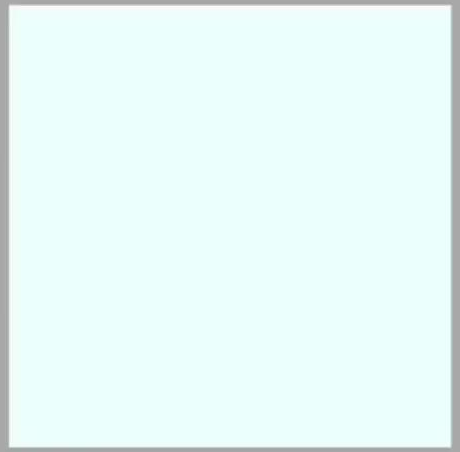
- OPTION ONE
- OPTION TWO
- OPTION THREE
- OPTION FOUR
  - PERFORM A TRICK AFTER ENTERING
    - SYNCHRONIZE BACKGROUND WITH SECONDARY ELEMENTS
      - DELETED ELEMENT 'EXPANDS FROM THE CENTER TO FILL THE SCREEN.'

Layers

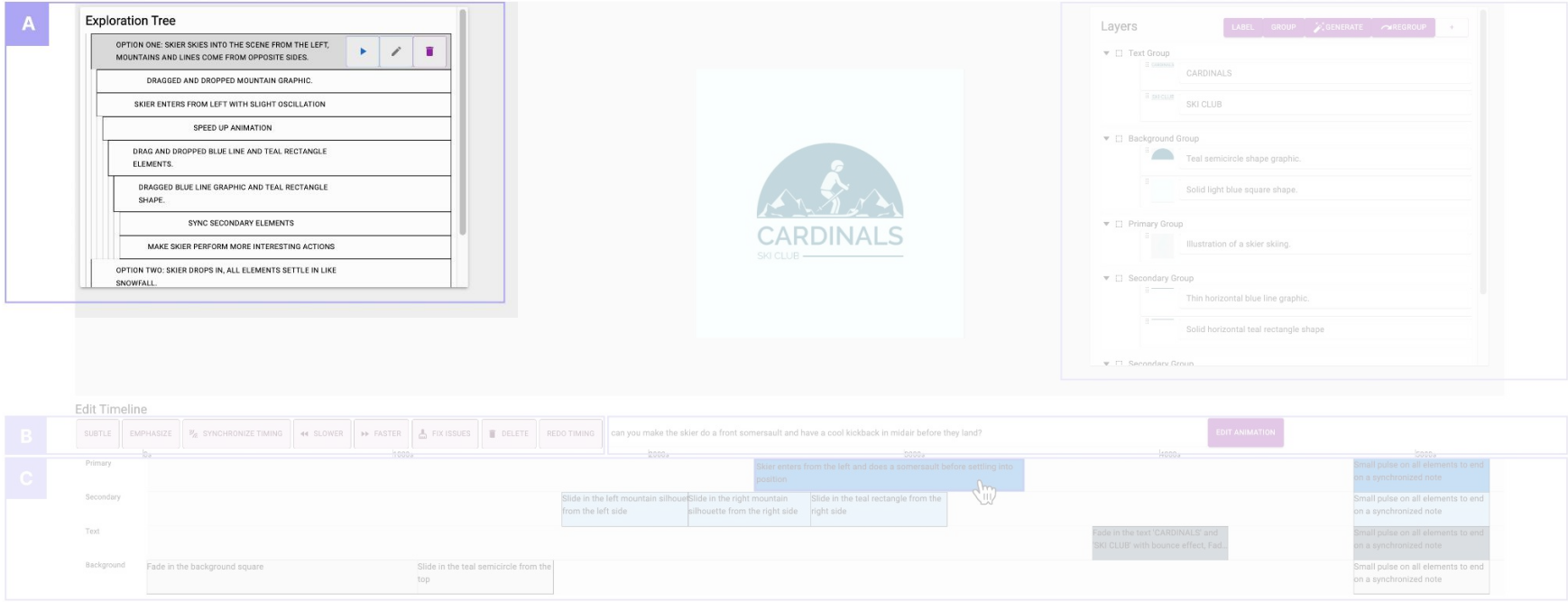
LABEL GROUP GENERATE REGROUP +

- Text Group
  - CARDINALS
  - SKI CLUB
- Background Group
  - Teal semicircle shape graphic.
  - Solid light blue square shape.
- Primary Group
  - Illustration of a skier skiing.
- Secondary Group
  - Thin horizontal blue line graphic.
  - Solid horizontal teal rectangle shape

Animation First Draft

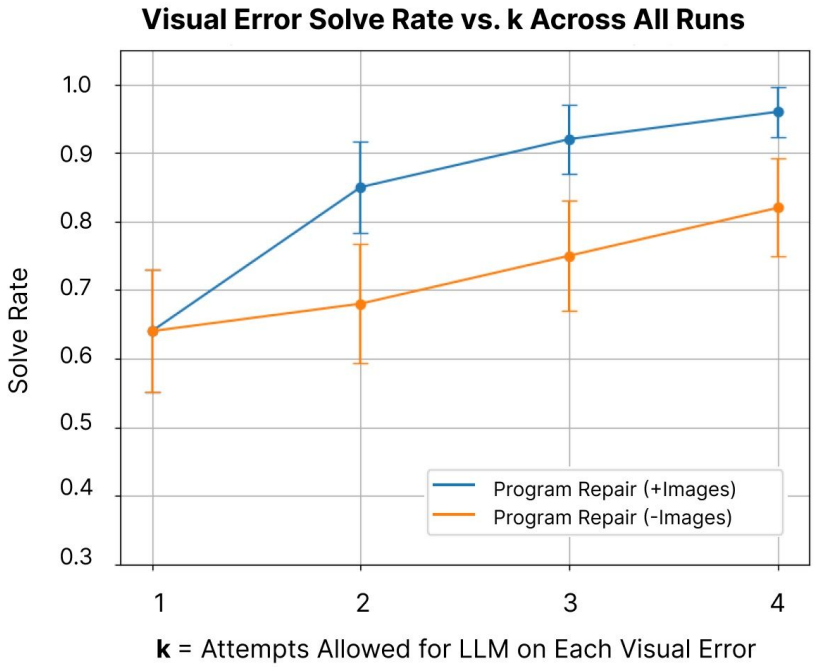


A version tree allows users to review and select animations.

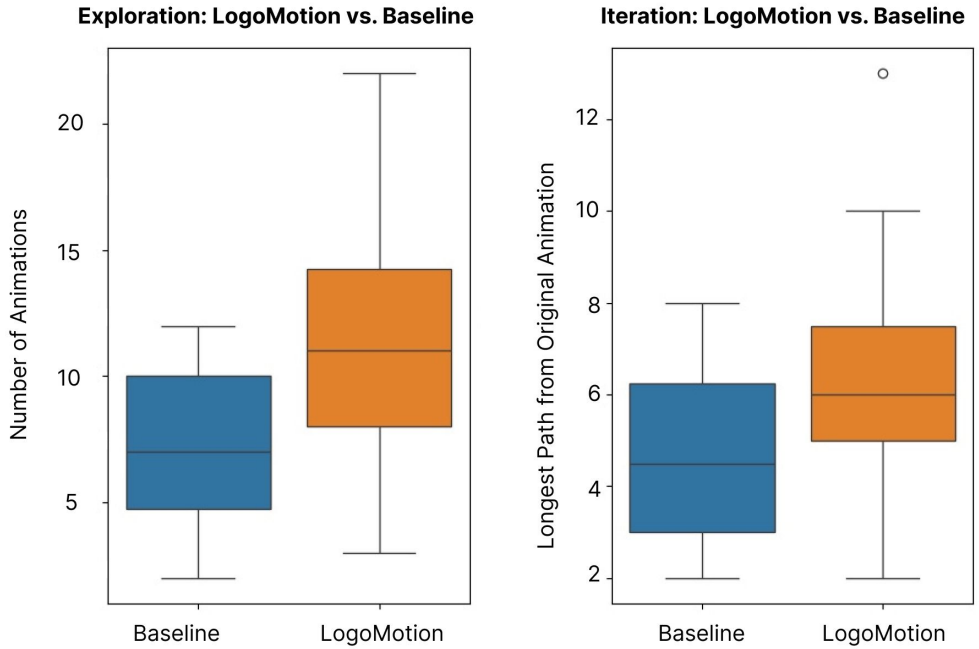


Animation Agents produce semantically meaningful animations.

Program Repair Enables a 95% Solve Rate.



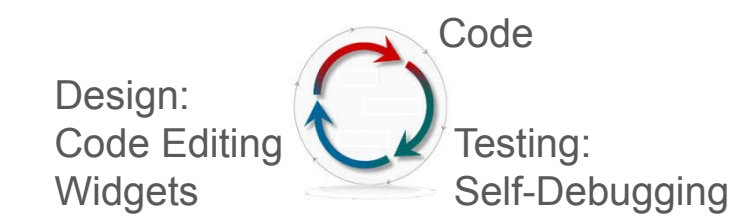
Editing Agents helped users explore more animations and craft better iterations than a baseline.

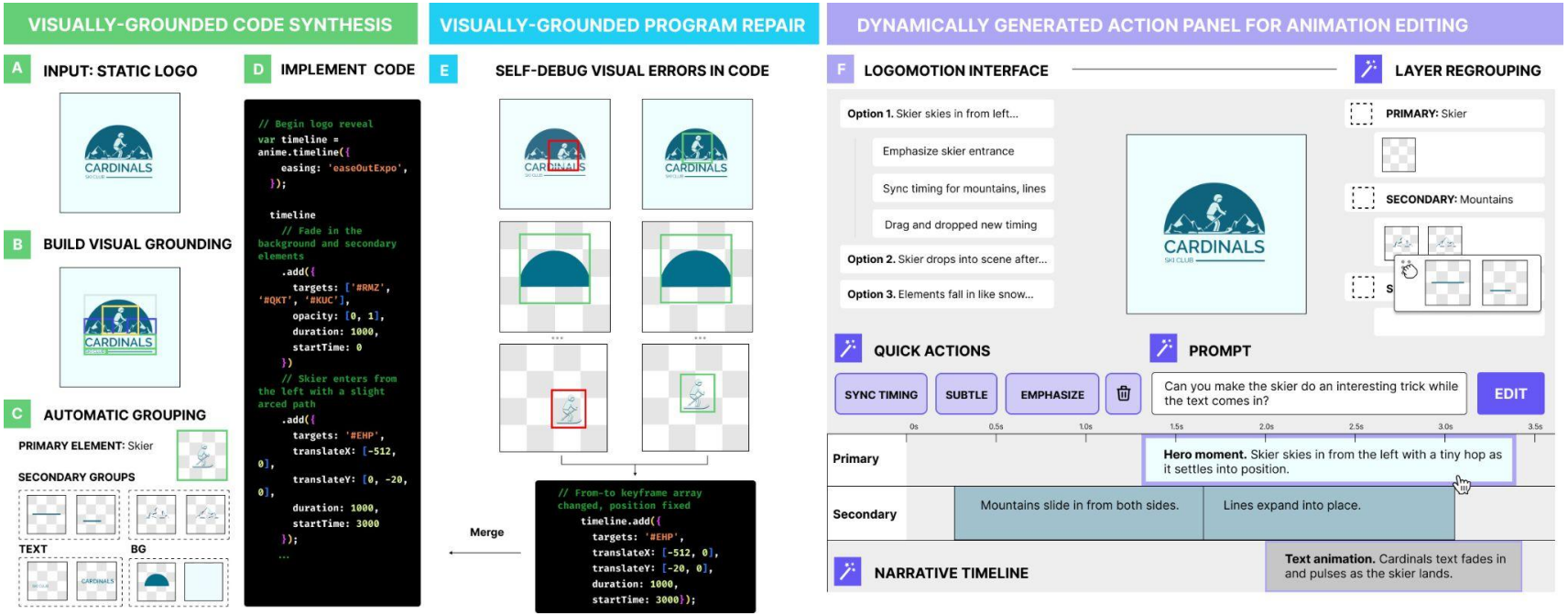


“It’s really cool especially for someone who couldn’t do it without this tool and wouldn’t spend a lot of time with Youtube videos or tutorials to do the very basics.” -P2

LogoMotion induces feedback with:

1. **Self Debugging**
2. **Code Editing Widgets**





Bridging Design and Code with Gen AI

Dynex

- Iterative Problem Specification
- Iterative Development

LogoMotion

- Self-debugging loop
- Iteration with AI Editing widgets

Double Agents

- Testing with user simulation
- Iterating on designs with AI suggestions

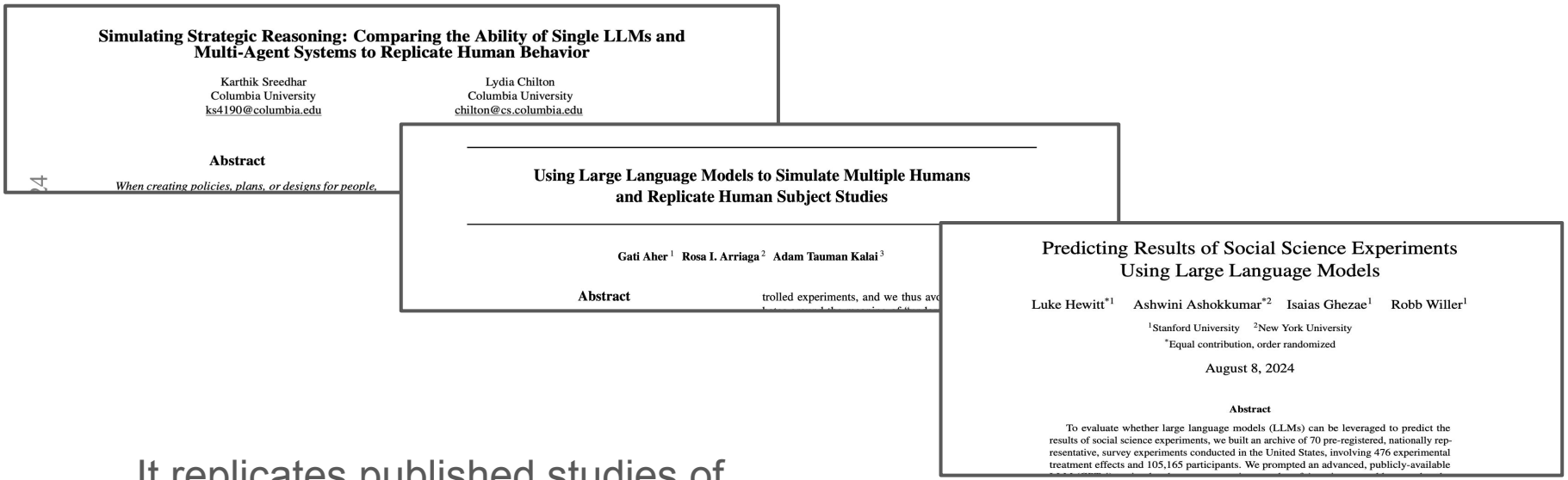
DoubleAgents

AI Agents for Task Completion and Human Simulation

Karthik Sreedhar, Alice Cai, Jenny Ma, Jeffrey V. Nickerson, Lydia B. Chilton.
Simulating Cooperative Prosocial Behavior with Multi-Agent LLMs: Evidence and Mechanisms for AI Agents to Inform Policy Decisions.
IUI 2025.

Tao Long, Xuanming Zhang, Sitong Wang, Jenny Ma, Karthik Sreedhar, Lydia B. Chilton
Double Agents. AI Agents for Task Completion and Human Simulation.
In preparation.

Several Studies have shown that AI can accurately replicate human behavior



Simulating Strategic Reasoning: Comparing the Ability of Single LLMs and Multi-Agent Systems to Replicate Human Behavior

Karthik Sreedhar
Columbia University
ks4190@columbia.edu

Lydia Chilton
Columbia University
chilton@cs.columbia.edu

Abstract

When creating policies, plans, or designs for people,

Using Large Language Models to Simulate Multiple Humans and Replicate Human Subject Studies

Gati Aher [1] Rosa I. Arriaga [2] Adam Tauman Kalai [3]

Abstract

trolled experiments, and we thus av

Predicting Results of Social Science Experiments Using Large Language Models

Luke Hewitt*[1] Ashwini Ashokkumar*[2] Isaias Ghezae[1] Robb Willer[1]

[1]Stanford University [2]New York University
*Equal contribution, order randomized

August 8, 2024

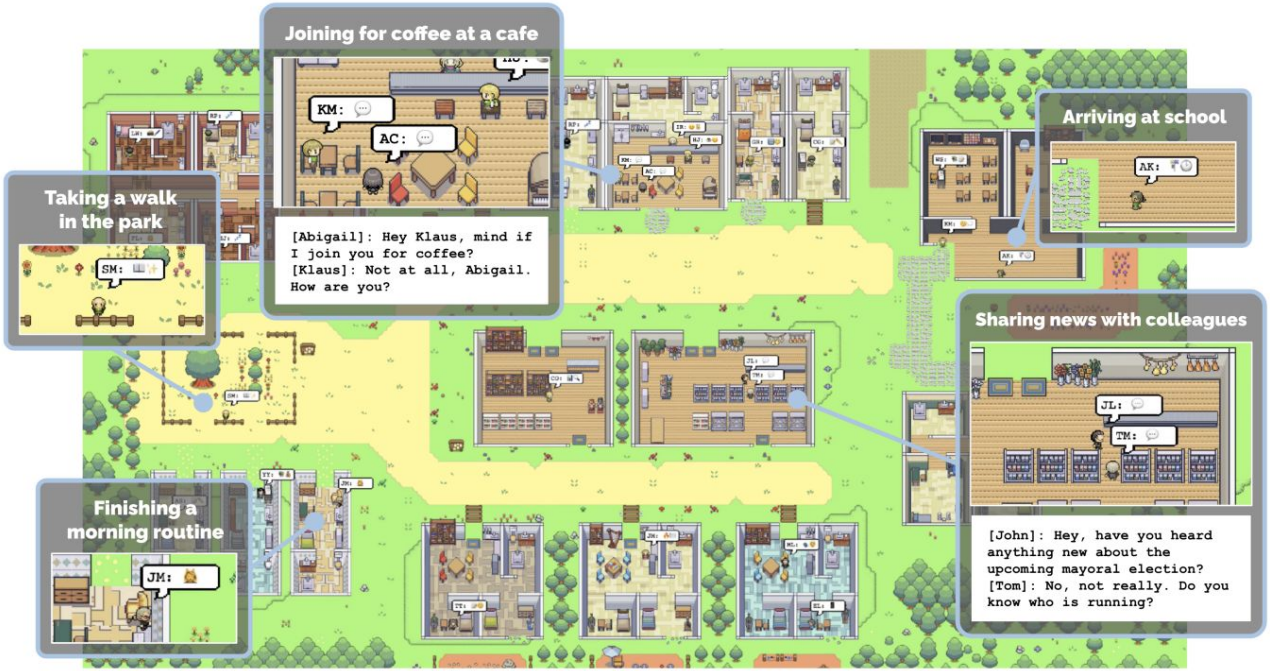
Abstract

To evaluate whether large language models (LLMs) can be leveraged to predict the results of social science experiments, we built an archive of 70 pre-registered, nationally representative, survey experiments conducted in the United States, involving 476 experimental treatment effects and 105,165 participants. We prompted an advanced, publicly-available

It replicates published studies of economic games

And 80% of of 70+ published and ***unpublished*** psychology games

Agents who “Observe”, “Think” and Act
can engage in “realistic” human activity, like organizing a party.



We use AI to accurately simulate
complex behavior outside of the lab

Classroom Simulation

There is one professor and 3 students:

The professor announces one of three late policies to the class ahead of a simulation:

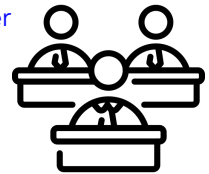
(1) harsh

2) some leniency

(3) very lenient

Classroom Simulation

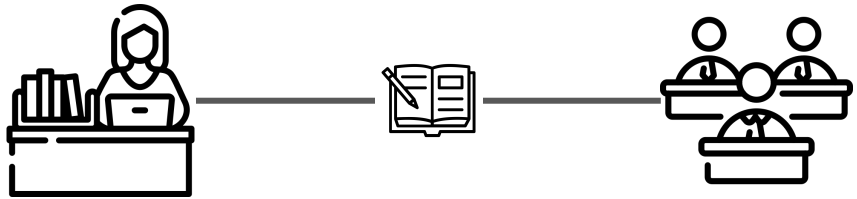
There is one professor and 3 students:



Students are assigned one of three "personality" traits.

Classroom Simulation

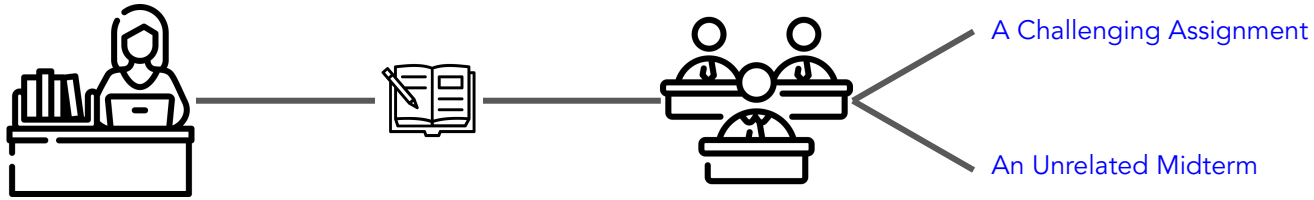
There is one professor and 3 students:



Professors assign weekly homework,
students work on it and turn it in.

Classroom Simulation

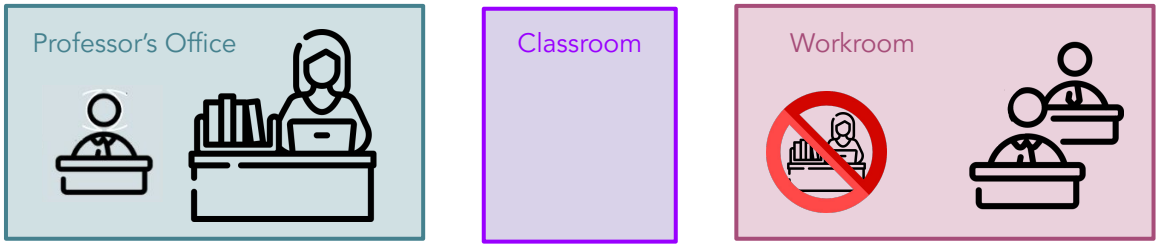
There is one professor and 3 students:



We also add stressors to the simulation to see what affects it has on behavior.

Classroom Simulation

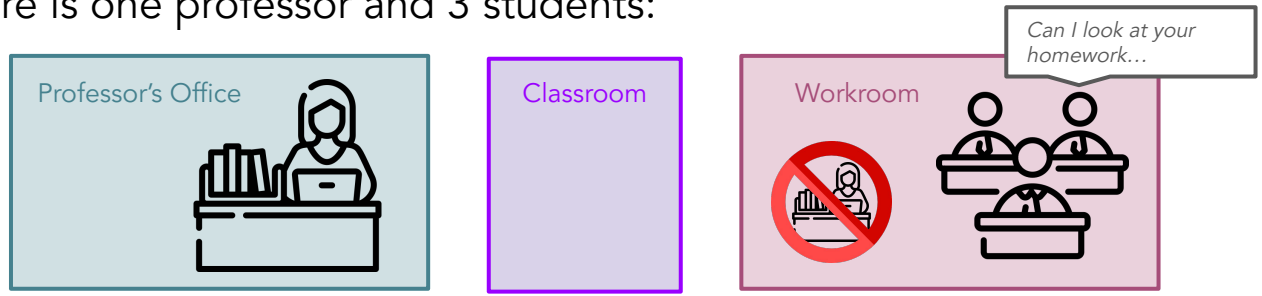
There is one professor and 3 students:



Students can enter the office to speak to or ask the professor questions.

When we add stressors, we see behaviors consistent with real students

There is one professor and 3 students:



1) **Cheating**. Procrastinators ask overachievers if they can "look at their homework"
2) Students email the professor pleading for extensions.

Since AI can simulate human behavior inside and outside the lab, it can provide human feedback to test software systems!

Simulating Strategic Reasoning: Comparing the Ability of Single LLMs and Multi-Agent Systems to Replicate Human Behavior

Karthik Sreedhar
Columbia University
ks4190@columbia.edu

Lydia Chilton
Columbia University
chilton@cs.columbia.edu

Using Large Language Models to Simulate Multiple Humans and Replicate Human Subject Studies

Gati Aher [1] Rosa I. Arriaga [2] Adam Tauman Kalai [3]

Predicting Results of Social Science Experiments Using Large Language Models

Luke Hewitt*[1] Ashwini Ashokkumar*[2] Isaias Ghezae[1] Robb Willer[1]

[1]Stanford University [2]New York University

*Equal contribution, order randomized

August 8, 2024

Abstract

To evaluate whether large language models (LLMs) can be leveraged to predict the results of social science experiments, we built an archive of 70 pre-registered, nationally representative, survey experiments conducted in the United States, involving 476 experimental treatment effects and 105,165 participants. We prompted an advanced, publicly-available

Simulating Cooperative Prosocial Behavior with Multi-Agent LLMs: Evidence and Mechanisms for AI Agents to Inform Policy Decisions

Karthik Sreedhar
ks4190@columbia.edu
Columbia University
New York, New York, USA

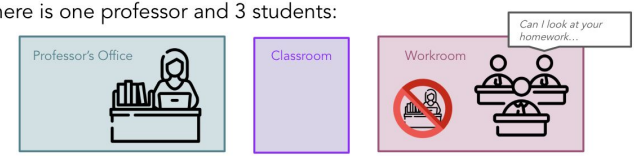
Alice Cai
acai@college.harvard.edu
Harvard University
Cambridge, Massachusetts, USA

Jenny Ma
jm5676@columbia.edu
Columbia University
New York, New York, USA

Jeffrey V. Nickerson
jnickers@stevens.edu
Stevens Institute of Technology
Hoboken, New Jersey, USA

Lydia B. Chilton
chilton@cs.columbia.edu
Columbia University
New York, New York, USA

There is one professor and 3 students:



1) Cheating. Procrastinators ask overachievers if they can "look at their homework"
2) Students email the professor pleading for extensions.

DoubleAgents

AI Agents for Task Completion and Human Simulation

Lydia Chilton
Columbia AI Summit, AI in Business

Human Coordination Problems (HCPs).

Warehouse.

Resource discovery.

Constraints. Availability.

Priority-based allocation.

Dynamic scheduling.

Optimization. Bipartite matching.

Real-time adjustments.

Uncertainty & adaptability

LLMs Agents drive real-world impact by taking action.

Tool / API / function calling to take actions.

GPT Operator / Claude Computer Use.

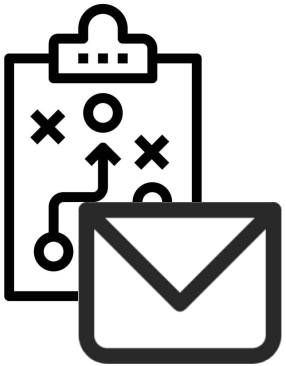
Orchestration

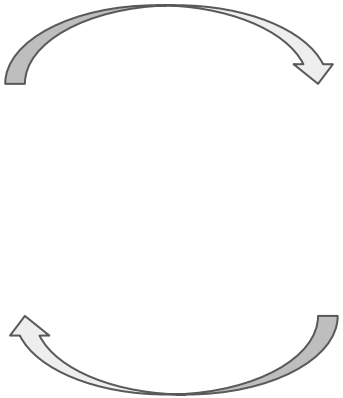
Workflow automation, task execution.

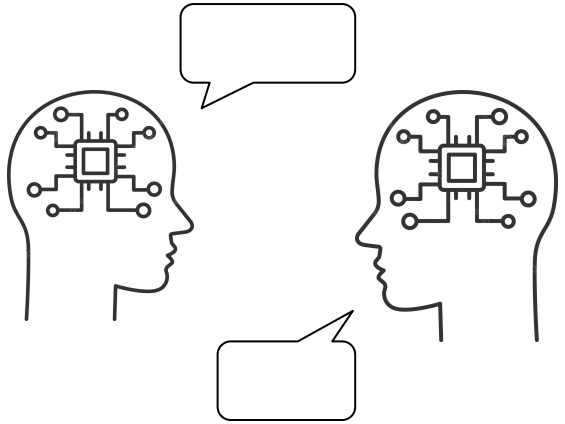
Human-in-the-loop systems & interfaces.

Action planning, execution, replanning…

DoubleAgents uses agents in a feedback loop to solve HCPs.







Agents **design** policies for

- Assigning work fairly
- Negotiating with workers' availability

Agents **Implement**

- Work Assignments
- Including contacting workers for availability questions and negotiation.

AI suggests **revised policies** when tests fail.

A human planner must approve new policies.

Simulations **Evaluate** Solutions

- Test the planner for multiple scenarios of human available.
- Vary human communication styles, responsiveness, etc.

Example scenario:

Speaker scheduling for research seminars.

Example scenario:
Speaker scheduling for research seminars.

1:1 relationships (A speaker can only speak at 1 seminar slot.)

Availability constraints, preferences, and priority availability

Varied response times (some people never respond, some reply instantly)

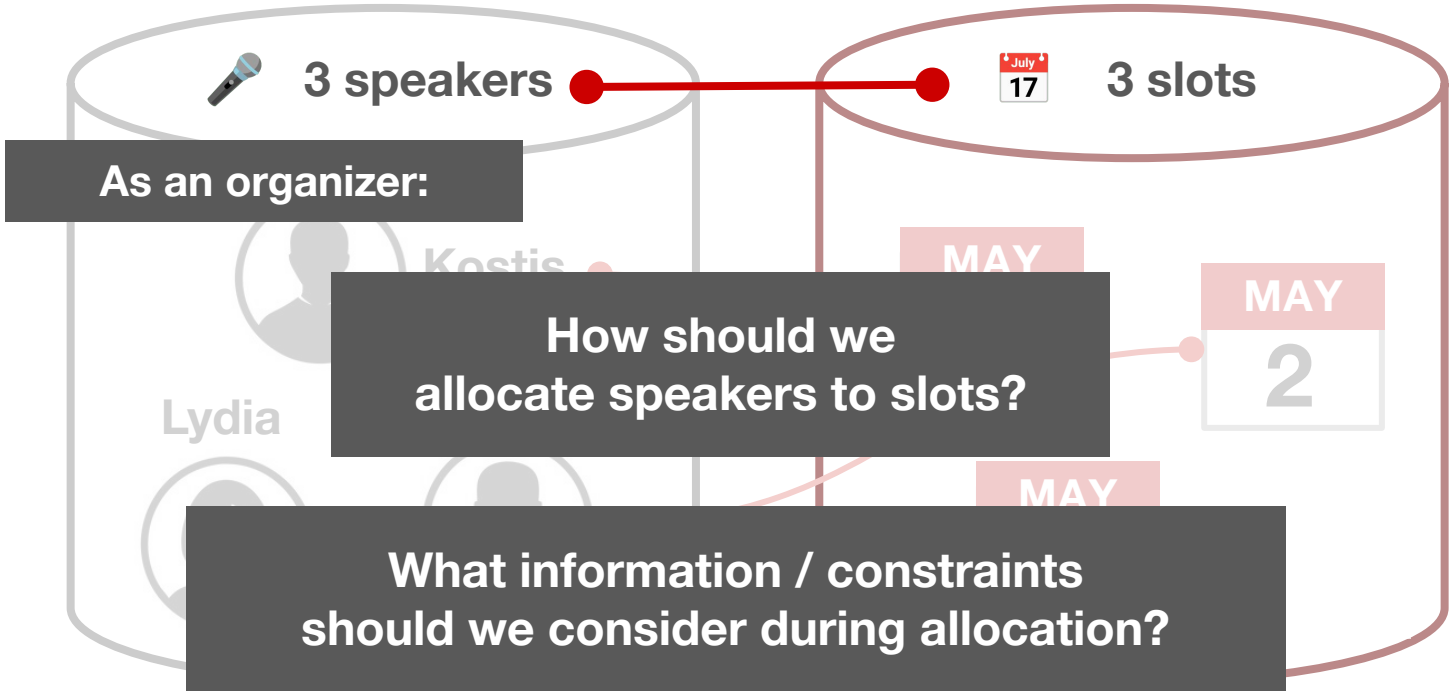
Extensive email communication (confirmation, request for additional availability, etc)

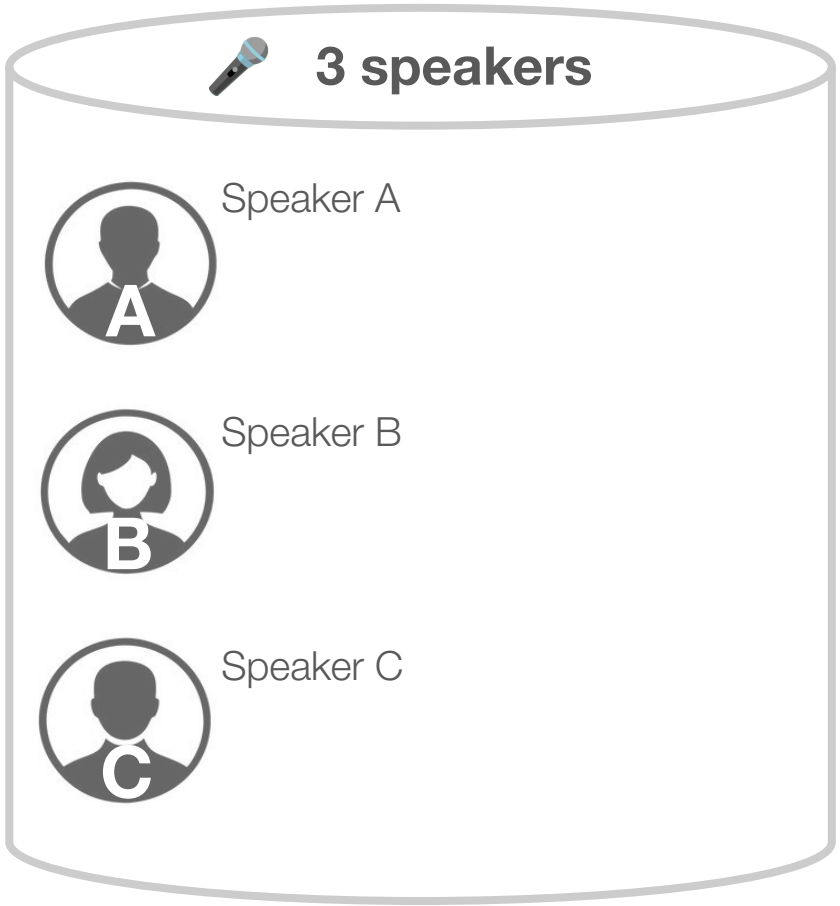
Multi-thread tracking email and sequential email follow-ups

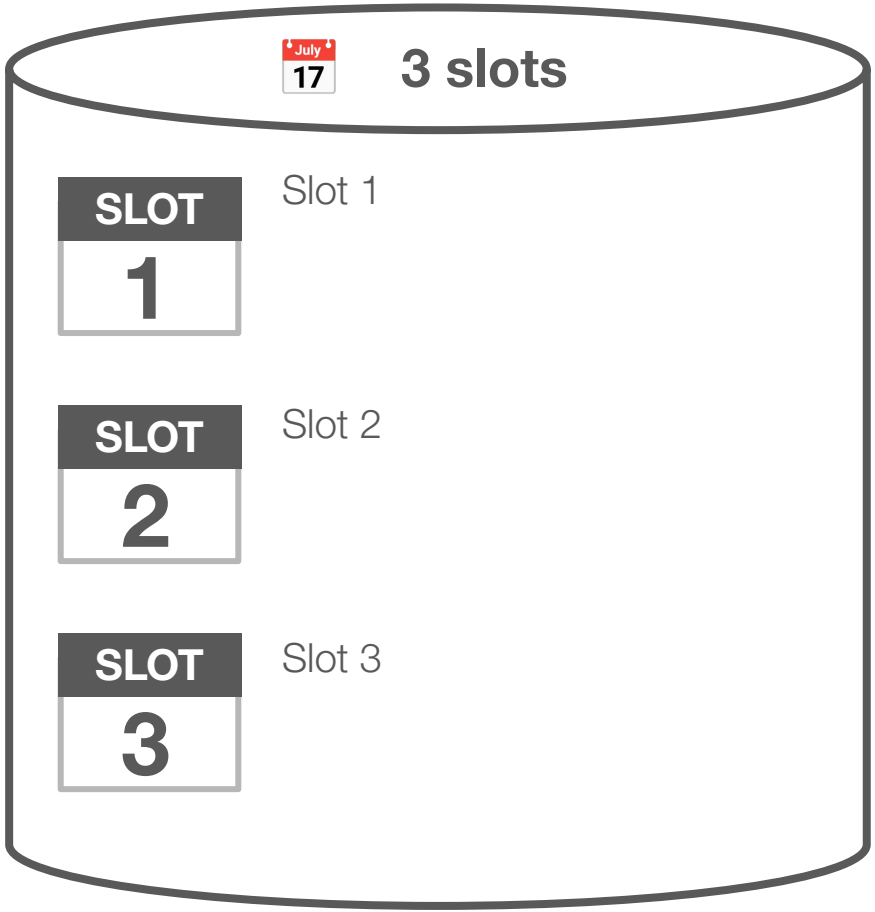
Example scenario:

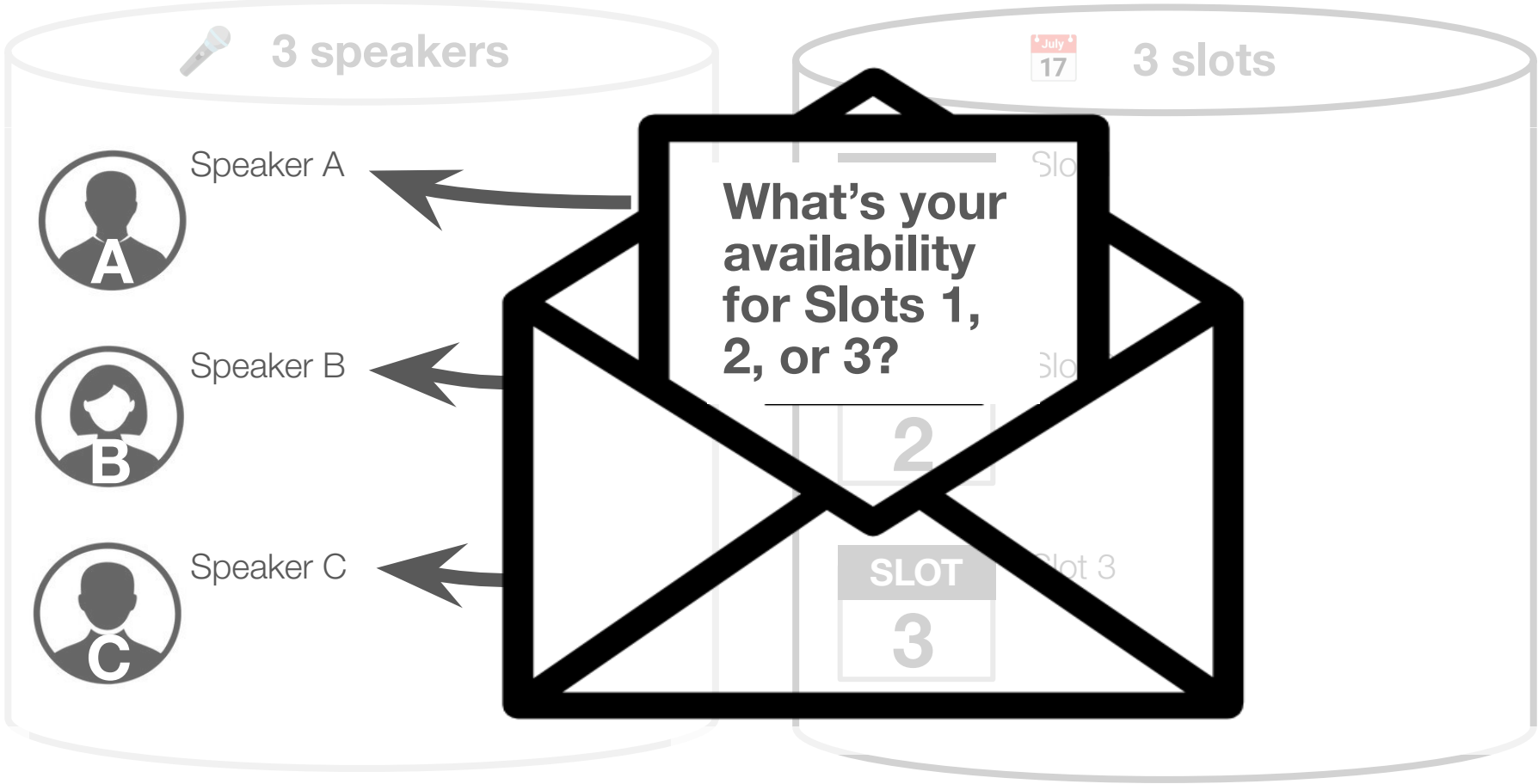
Speaker scheduling for research seminars.

Example:



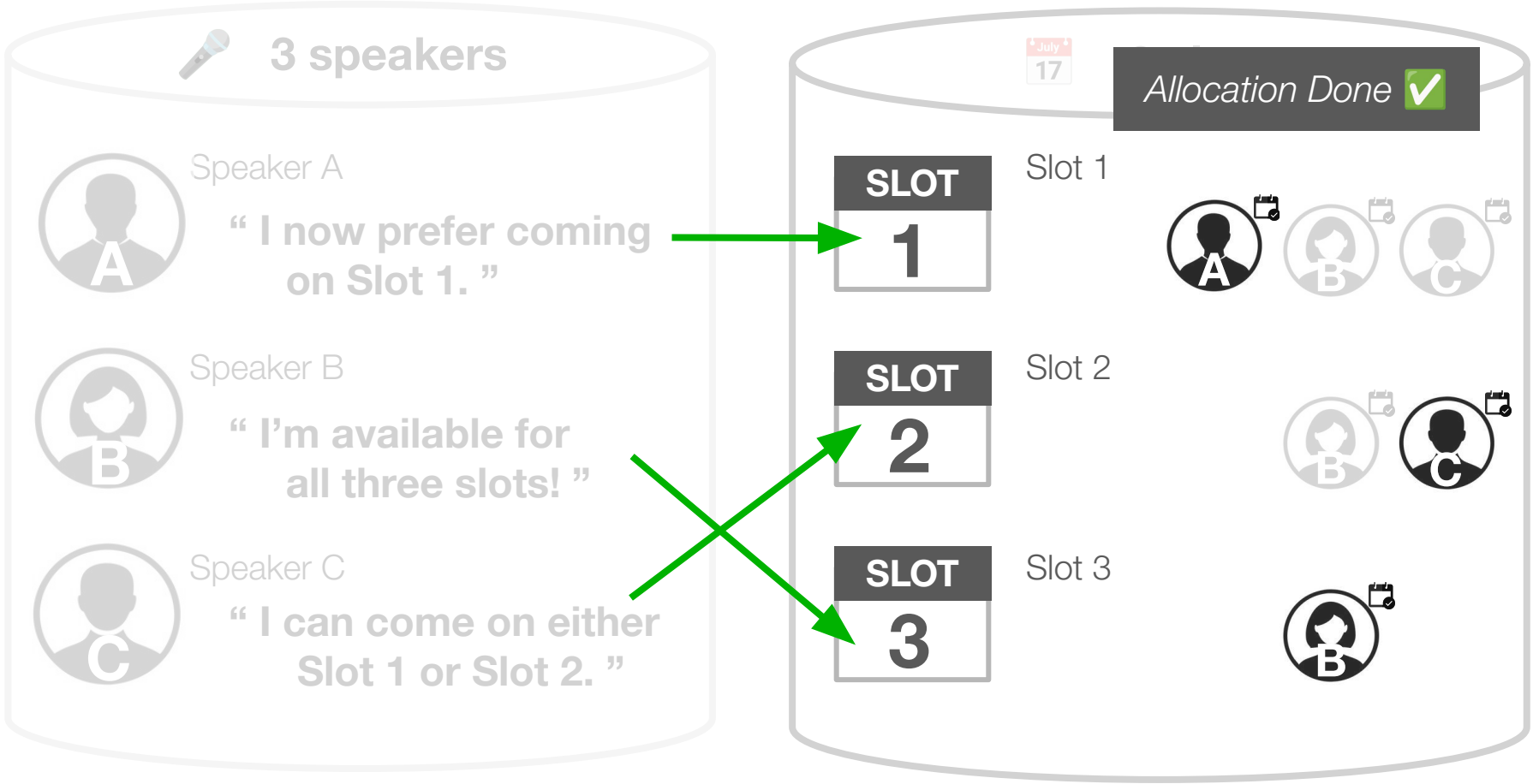

3 speakers
A
Speaker A
B
Speaker B
C
Speaker C


July
17
3 slots
SLOT
1
Slot 1
SLOT
2
Slot 2
SLOT
3
Slot 3

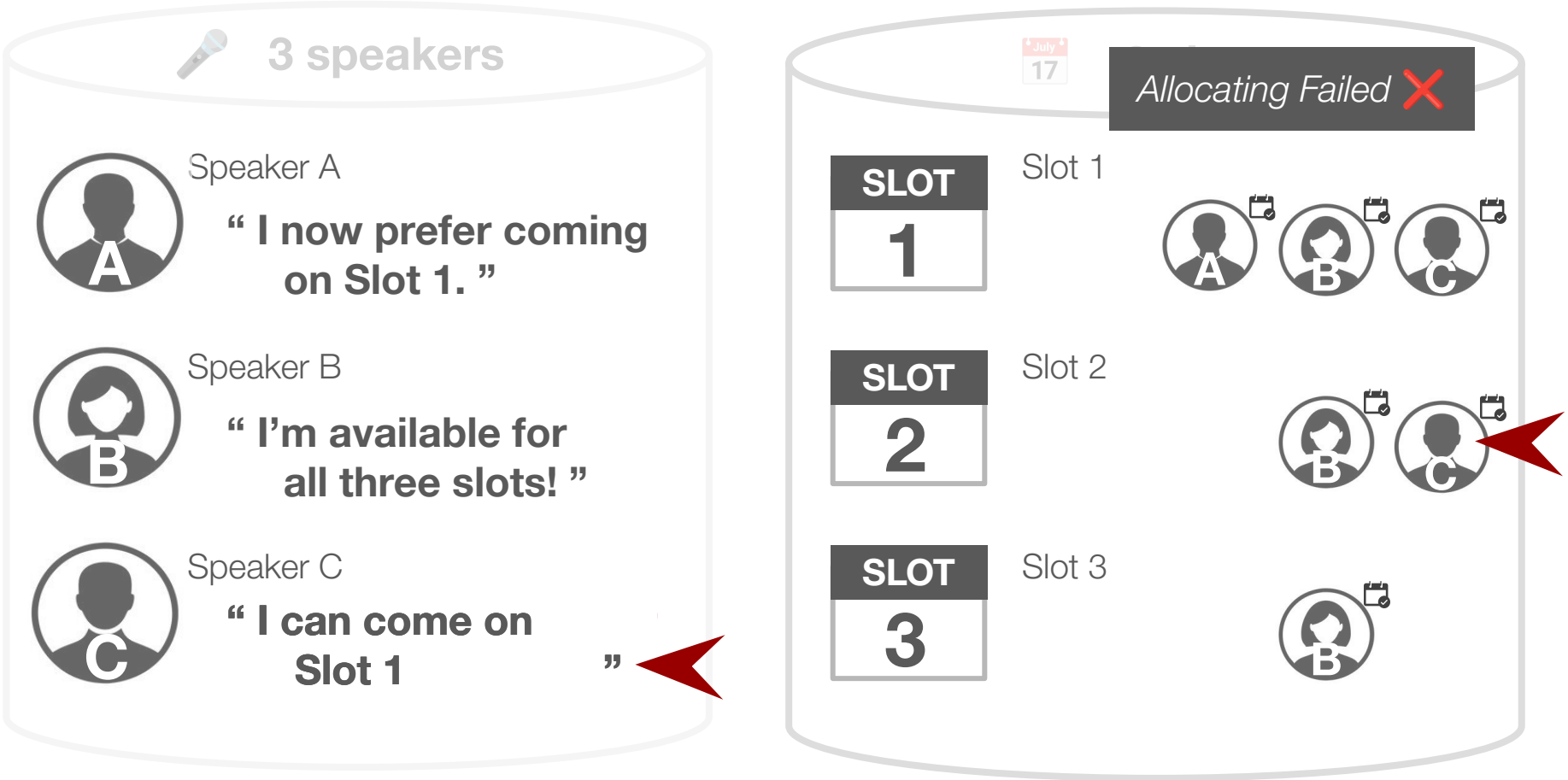

3 speakers
3 slots
Speaker A
Speaker B
Speaker C
A
B
C
What's your availability for Slots 1, 2, or 3?
July
17
2
SLOT
3

🟩 **Easy Case:** No conflicted availability

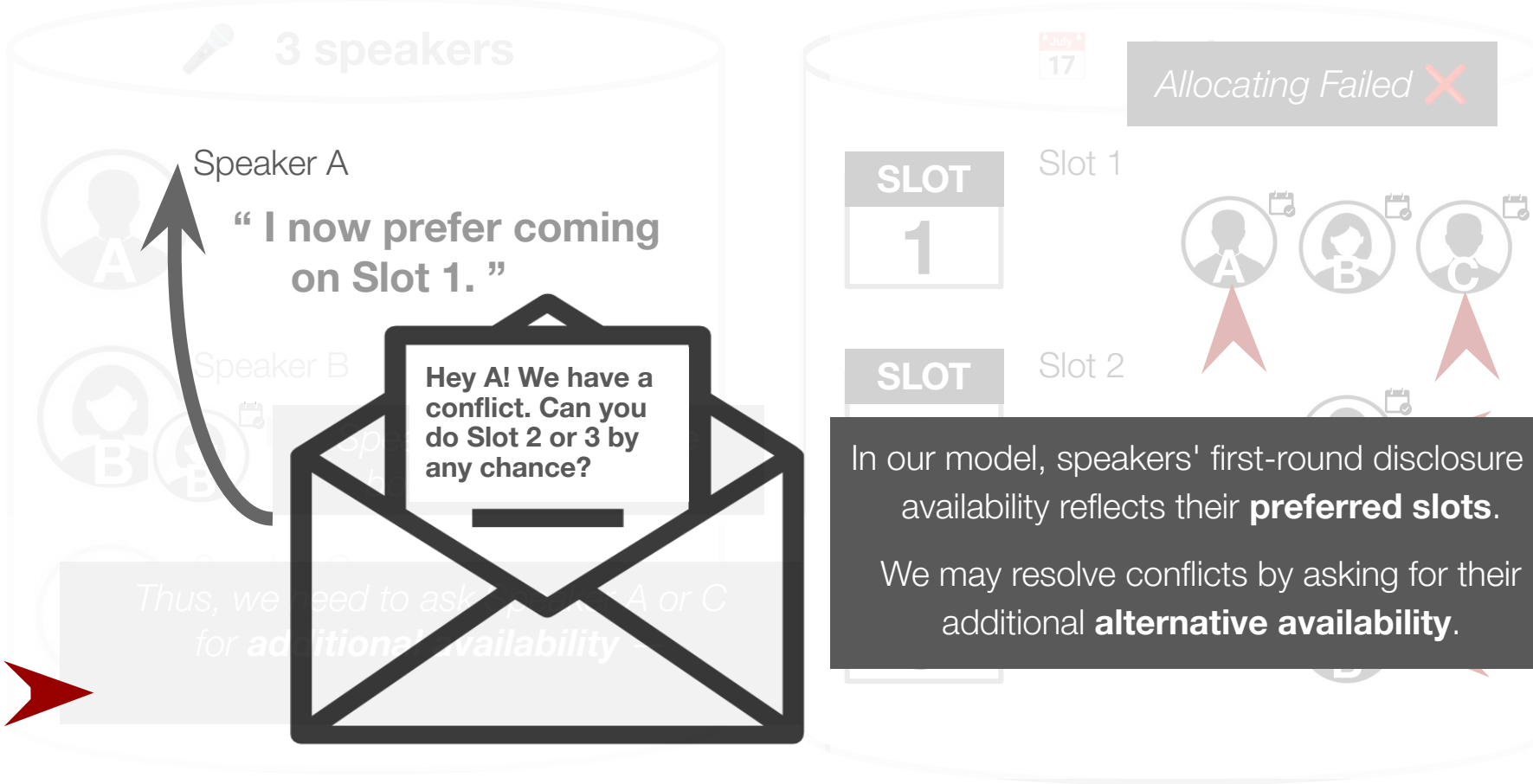
🟩 **Easy Case:** No conflicted availability



🟨 **Medium Case:** Conflicted availability



🟨 **Medium Case:** Conflicted availability - must update policies



🟨 **Medium Case:** Conflicted availability - must update policies

📧 *Email Reply Received!*

Speaker A

“ I now prefer coming on Slot 1. ”

“ I see, don't worry! I can do Slot 3 if that works better for you.”

Speaker B

“ I’m available for all three slots! ”

Speaker C

“ I can come on Slot 1. ”

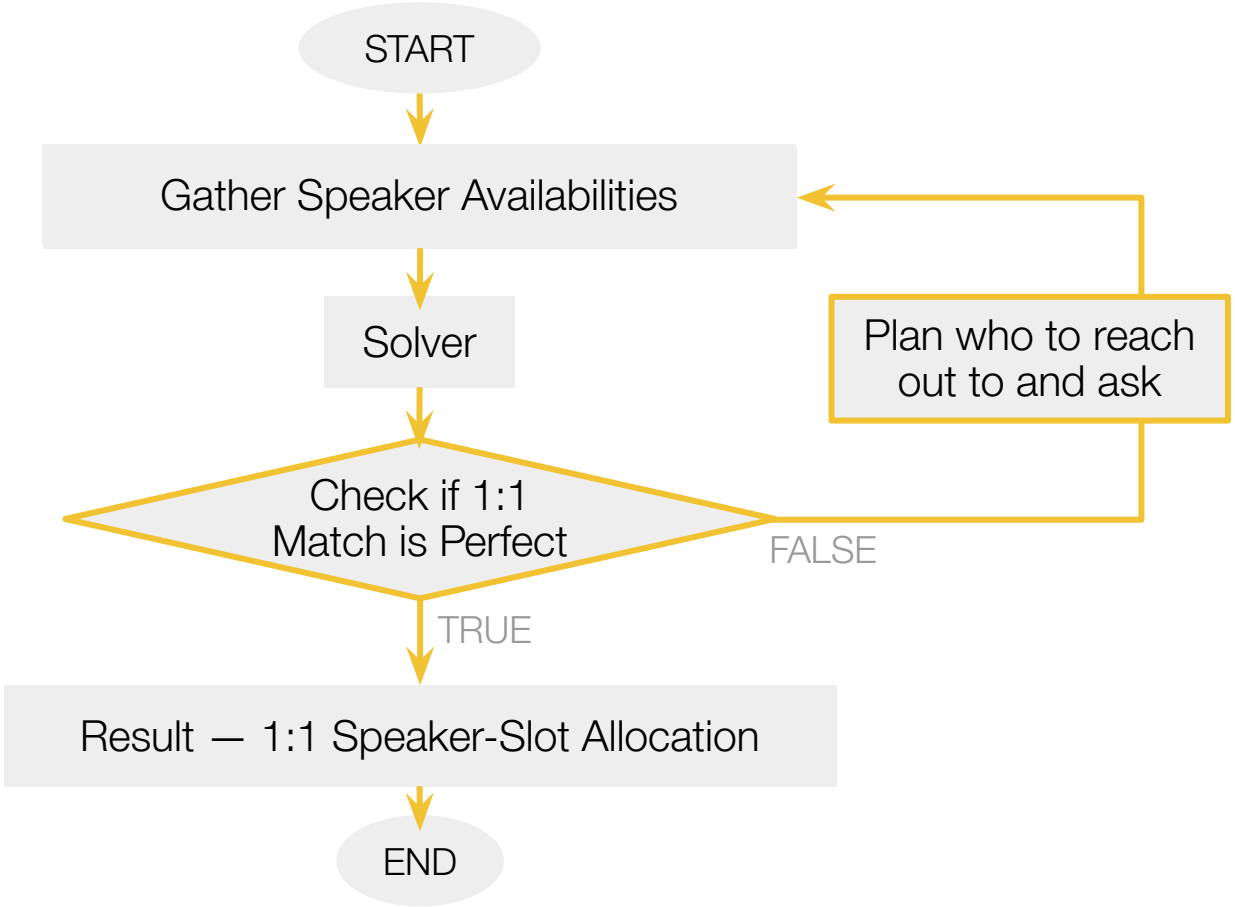
Allocation Done ✅

SLOT 1	Slot 1
SLOT 2	Slot 2
SLOT 3	Slot 3

Design (Policies):

- The solver can reach out to speakers again to ask for additional availability.

Execution Agent: HCP Solver



🟥 Repeat for harder test cases to discover more policies (designs) needed for real world deployment.

Simulated test cases	Policies to establish
1. Workers don't have enough availability.	● How many times is it okay to ask workers for more availability?
2. Workers who don't respond to emails	● How long should you wait to nudge? ● How many nudges should you do without a response? ● How you you balancing assigning dates to prompt responders versus hounding non-responders?

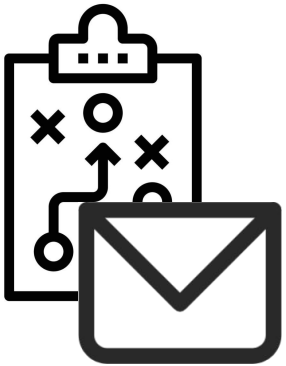
DoubleAgents can solve all test cases, following new policies

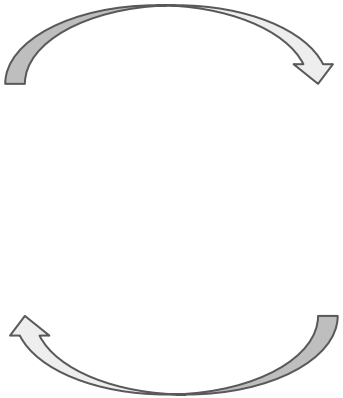
	🟩 **Easy:** No conflicts	🟨 **Medium:** Few conflicts	🟥 **Hard:** Lots of conflicts
Is the allocation correct? The final speaker-slot allocation matches speakers agents' availabilities.	**100%**	**100%**	**100%**
# rounds of plan execution needed for achieving a perfect allocation			

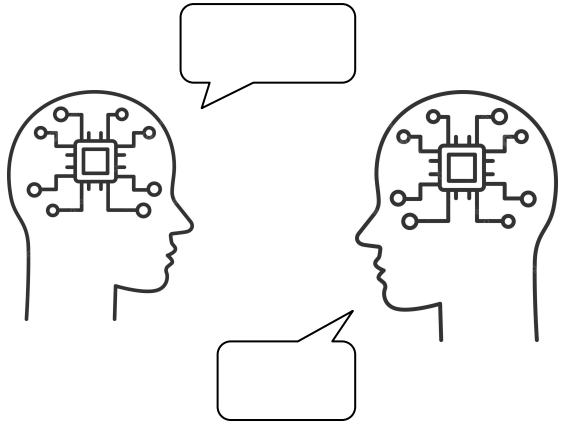
DoubleAgents can solve all test cases, following new policies

	🟩 **Easy:** No conflicts	🟨 **Medium:** Few conflicts	🟥 **Hard:** Lots of conflicts
Is the allocation correct? The final speaker-slot allocation matches speakers agents' availabilities.	**100%**	**100%**	**100%**
# rounds of plan execution needed for achieving a perfect allocation	**0**	**1.67**	**3.38**

DoubleAgents uses agents in a feedback loop to solve HCPs.







Agents **design** policies for

- Assigning work fairly
- Negotiating with workers' availability

Agents **Implement**

- Work Assignments
- Including contacting workers for availability questions and negotiation.

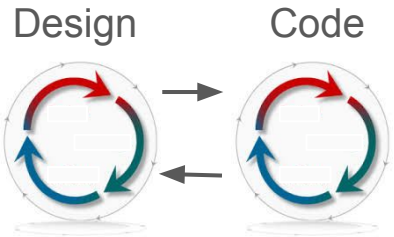
AI suggests **revised policies** when tests fail.

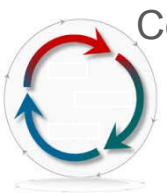
A human planner must approve new policies.

Simulations **Evaluate** Solutions

- Test the planner for multiple scenarios of human available.
- Vary human communication styles, responsiveness, etc.

Bridging Design and Code with Gen AI





Dynex

- Iterative Problem Specification
- Iterative Development

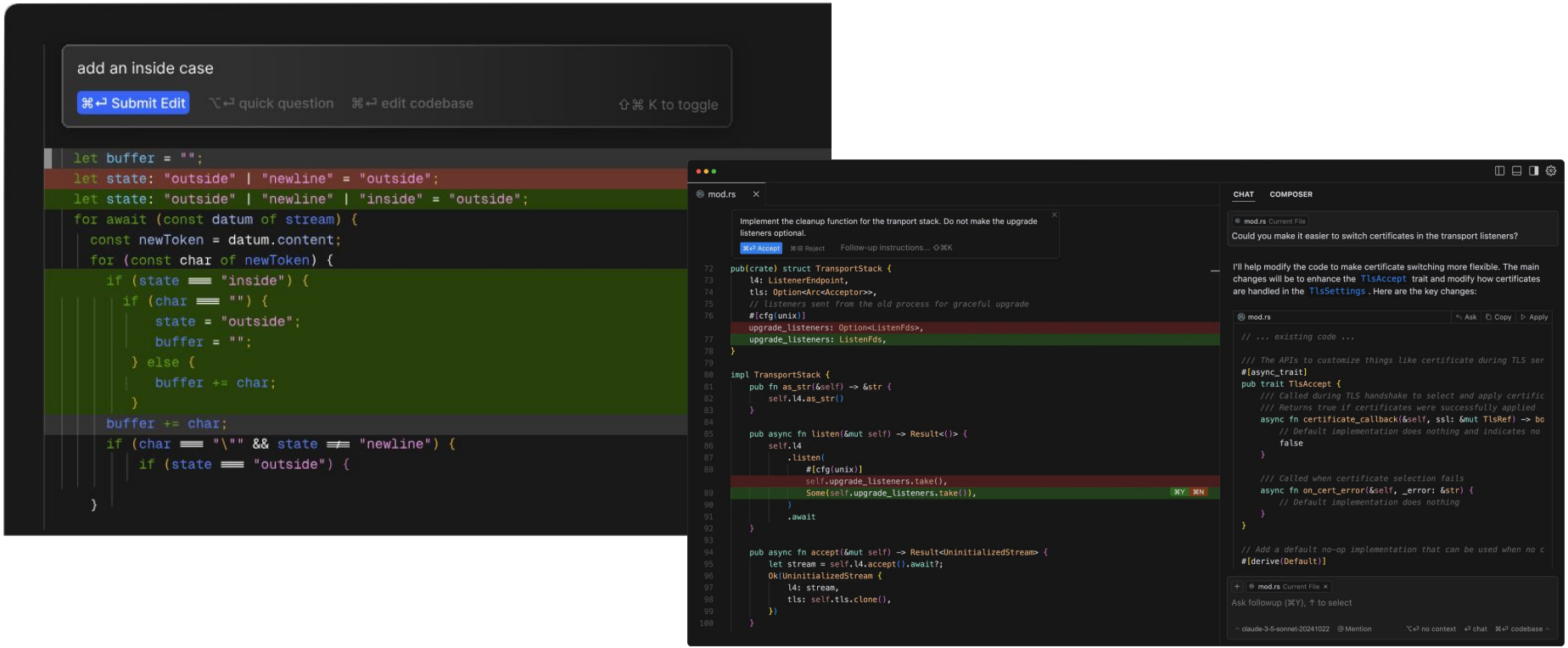
LogoMotion

- Self-debugging loop
- Iteration with AI Editing widgets

Double Agents

- Testing with user simulation
- Iterating on designs with AI suggestions

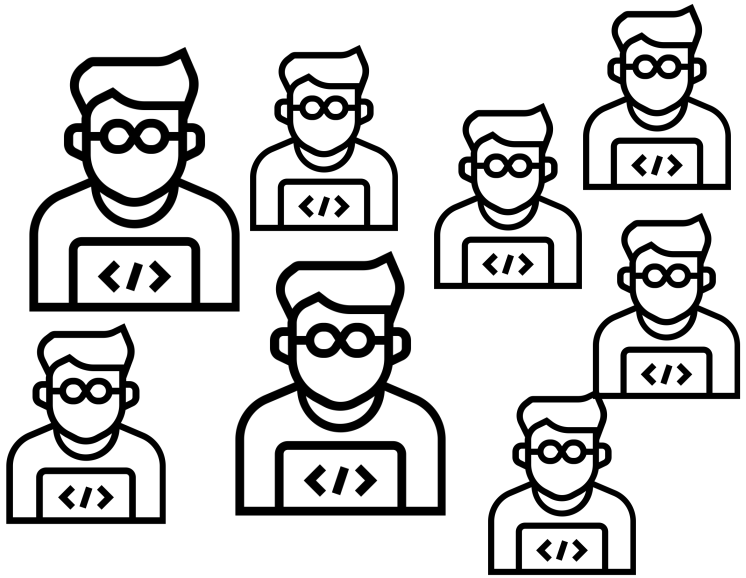
AI Code generation has transformed the way programmers work.



Design
(What do we build?)

Created by Brickclay
from Noun Project

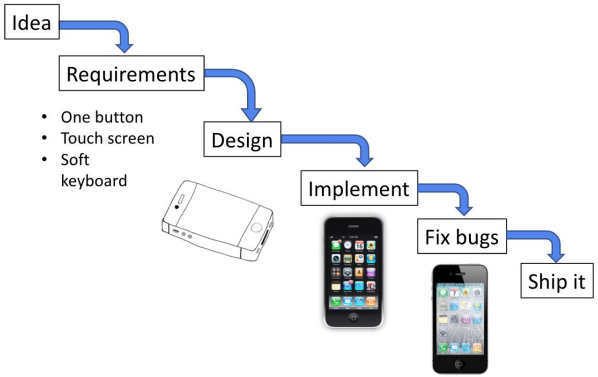
Engineering
(How do we build it?)



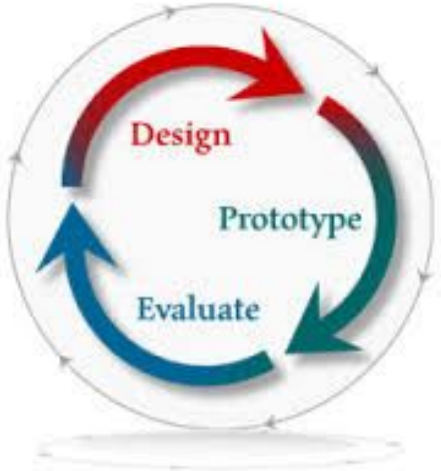
Created by WiStudio
from Noun Project

Two Software Design Processes

Feedforward

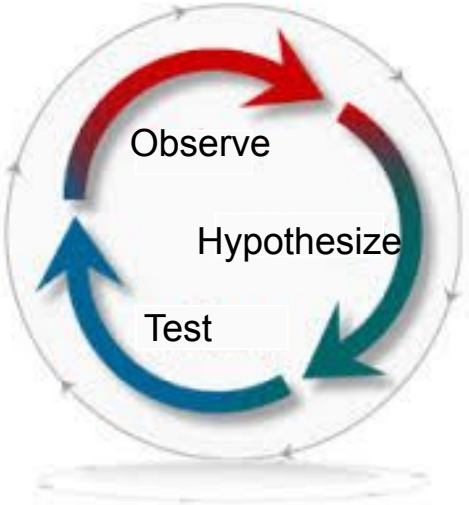


Feedback

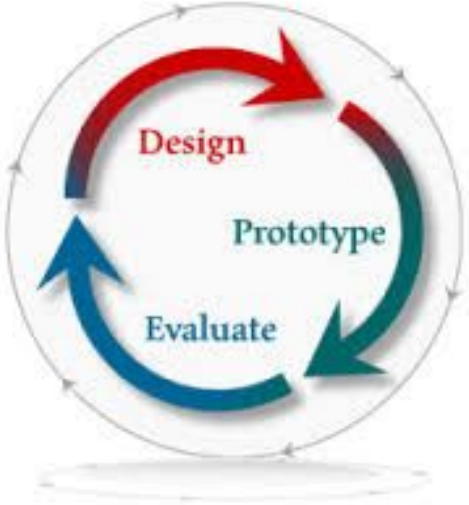


Human-Centered Design: Two feedback loops

Person with a challenge



Ideas

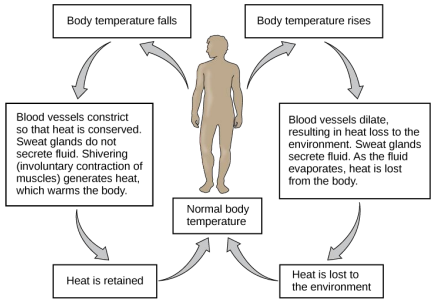


Product

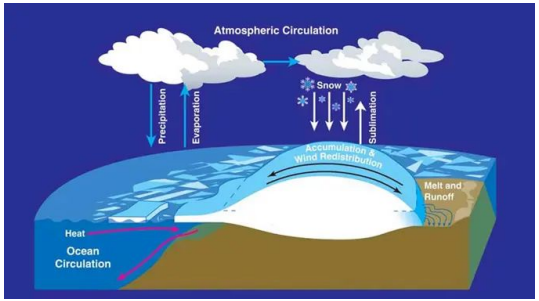
Understand the problem

Solve the problem

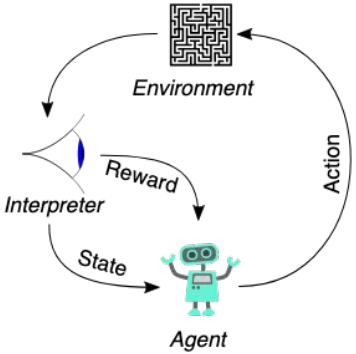
Feedback loops are essential to all systems



The human body



Earth systems



Training AI

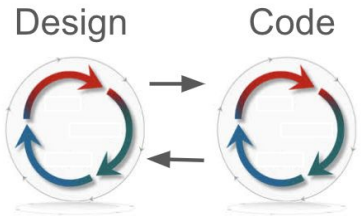
Education

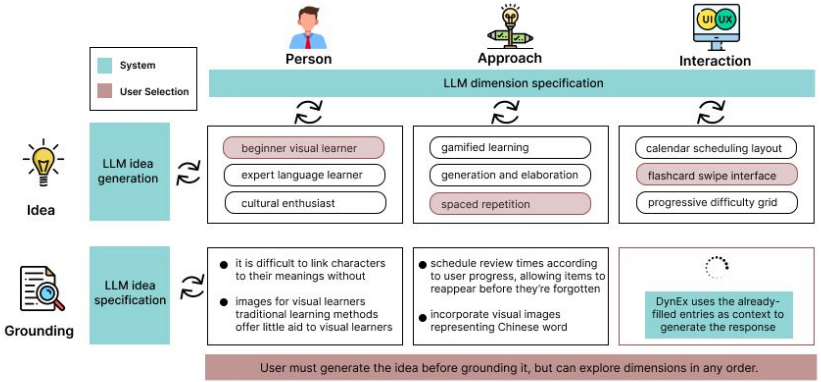
Childhood Development

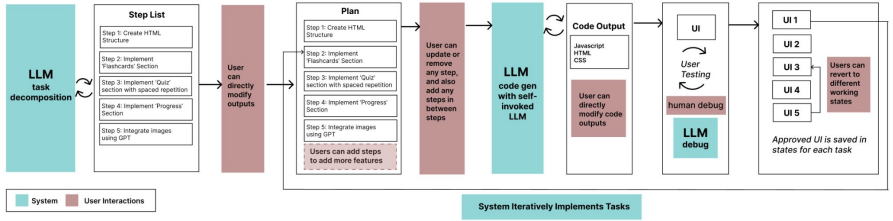
AI Code generation isn’t enough to create better software products…

We have to close feedback loops between design and engineering.

Dynex: Bridging Design and Code





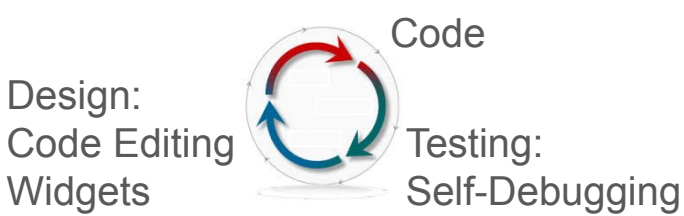


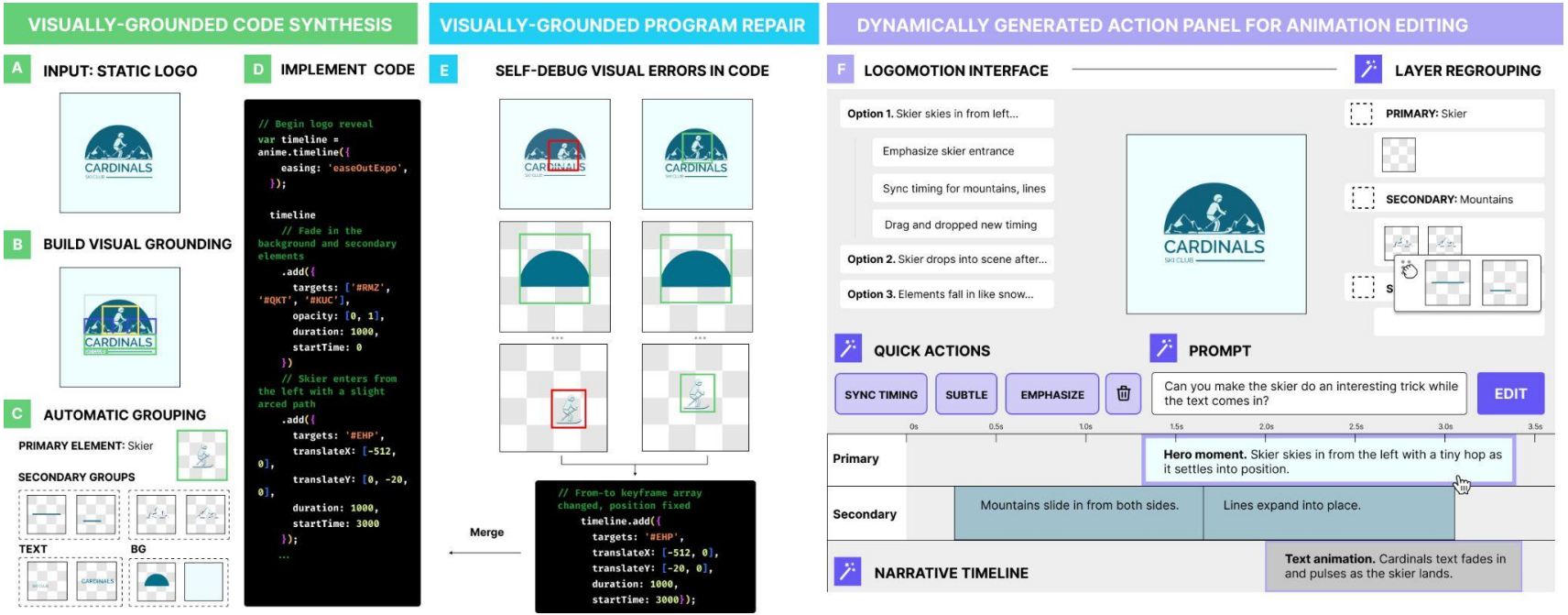
Iterative Problem Specification (Design)

Modular Code Generation and Testing

LogoMotion induces feedback with:

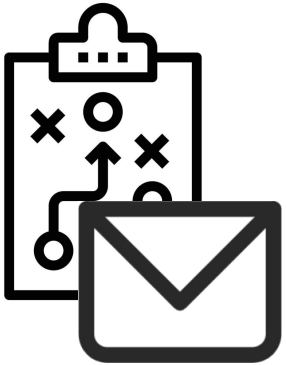
1. Self Debugging
2. Design iteration w/ AI Code Editing Widgets

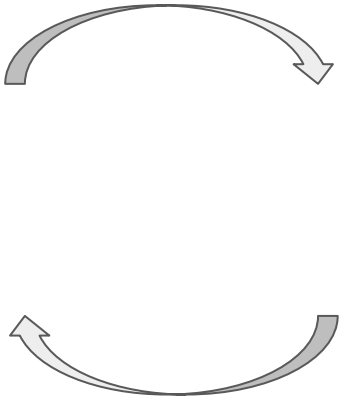


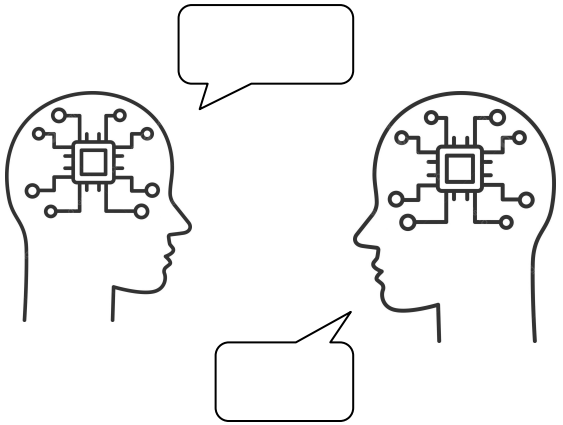


DoubleAgents uses feedback for

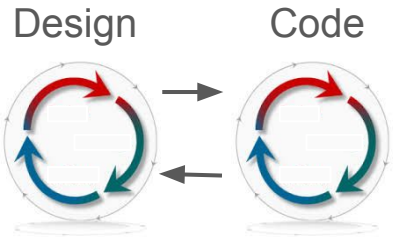
1) Simulating user testing
2) Policy (Design) Updates

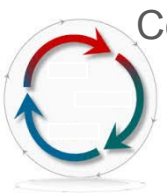






Bridging Design and Code with Gen AI





Dynex

- Iterative Problem Specification
- Iterative Development

LogoMotion

- Self-debugging loop
- Iteration with AI Editing widgets

Double Agents

- Testing with user simulation
- Iterating on designs with AI suggestions