

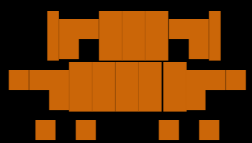
co-active ai-assisted programming

Nadia Polikarpova

UCSC CSE Colloquium – March 11, 2026

UC San Diego
COMPUTER SCIENCE & ENGINEERING

invisible decisions



Claude Code v2.1.69
Opus 4.6 · Claude Max
~/aithoscope

- > Hey Claude! Help me port this pipeline from local CSV storage to BigQuery
- Let me first explore the system...
- Okay now I have a complete understanding!
...
Here is my plan. Want me to implement this?
- > Looks legit, go for it!

this is almost right but...

... joins without partition pruning are expensive!

```
...
query = f"""
SELECT p.feed_id, p.message_id, p.text
FROM `{FILTERS_TABLE}` f
JOIN `{POSTS_TABLE}` p ON p.post_pk = f.post_pk
WHERE f.filter = 'logreg'
      AND f.pass = TRUE
      AND f.customer_id = {customer_id}
      AND NOT EXISTS (
        SELECT 1 FROM `{DECISIONS_TABLE}` d
        WHERE d.post_pk = f.post_pk
              AND d.customer_id = {customer_id}
      )
"""
...
```

... it's also a mess of SQL strings

problem: key decisions aren't
visible or steerable



85%

use or plan to use AI
for development*



51%

professional developers
use it daily

*2025 StackOverflow survey

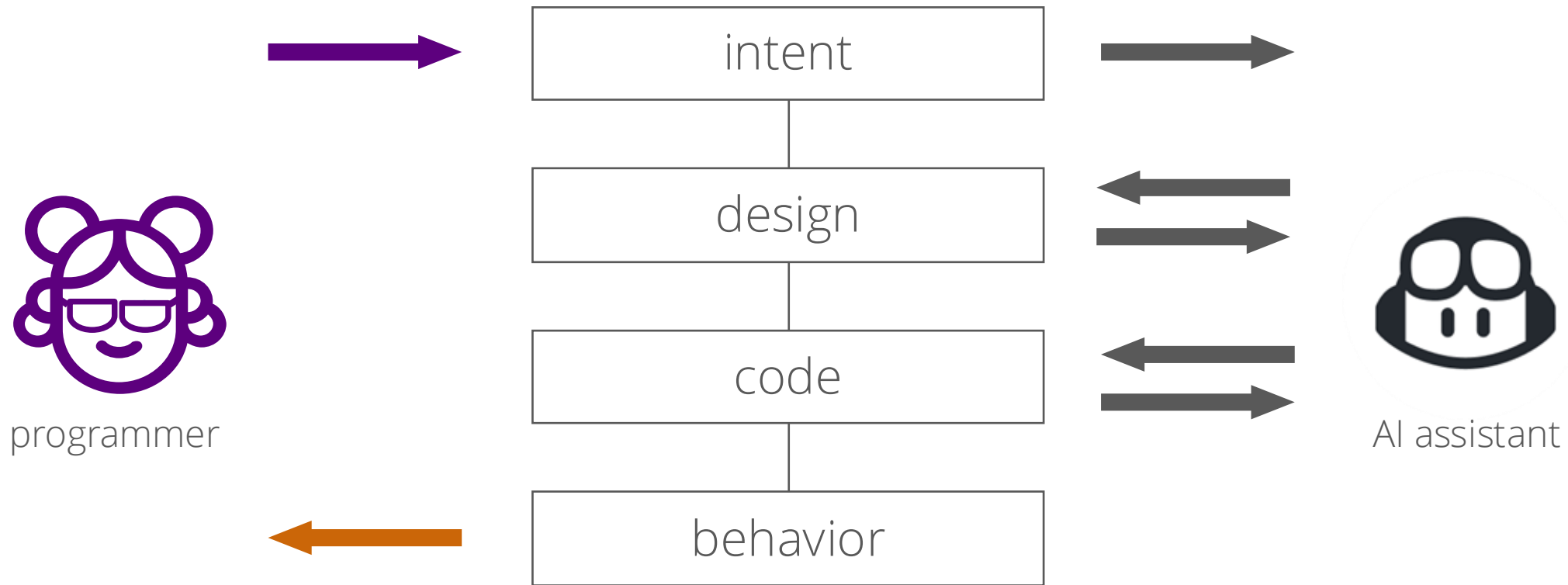


most common frustration:

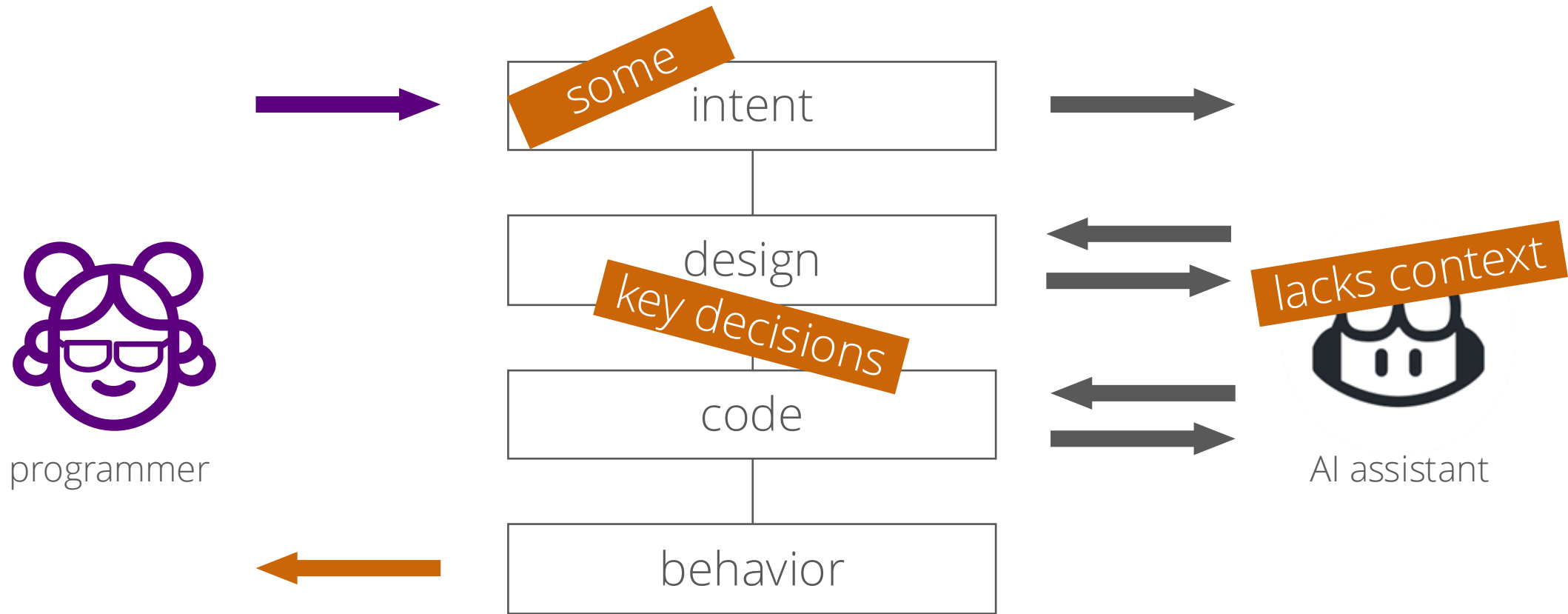
solutions that are almost right
but not quite

*2025 StackOverflow survey

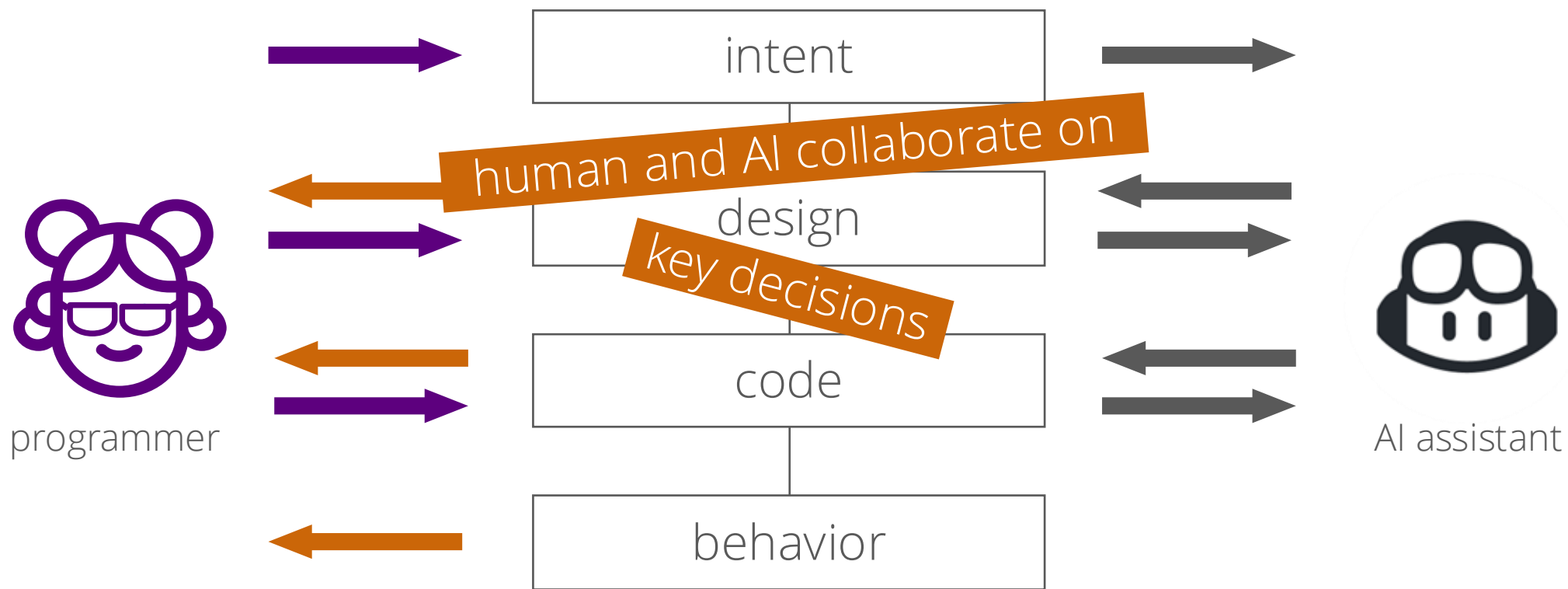
automated ai-assisted programming



~~automated~~ ai-assisted programming



co-active ai-assisted programming



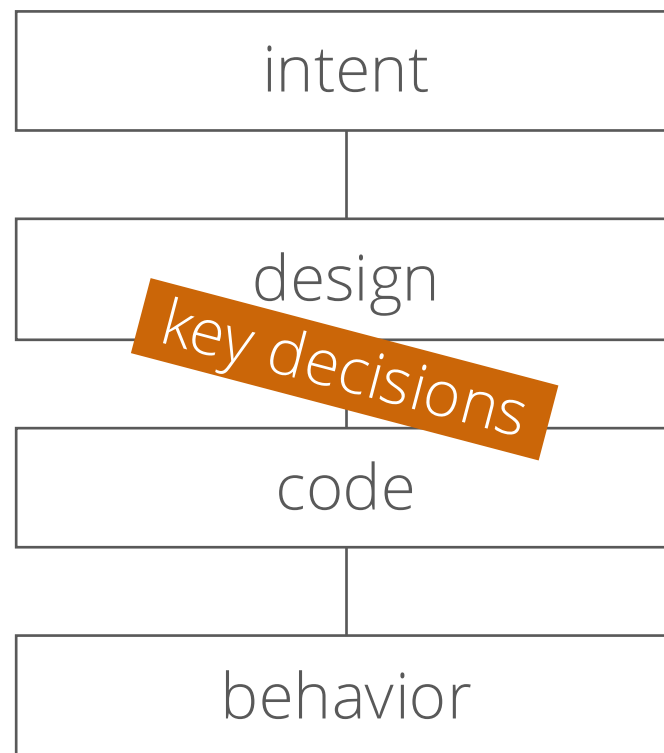
co-active ai-assisted programming



observability
user is aware of decisions



controllability
user can steer decisions



co-active ai-assisted programming

aporia: decision-oriented programming with agents
[in progress]



design



hilde: intentional code generation via human-in-the-loop decoding
[VL/HCC'25]



code

leap: validating AI-generated code with live programming
[CHI'24]



behavior

this talk

› **leap**: validating AI-generated code with live programming
[CHI'24]

hilde: intentional code generation via human-in-the-loop decoding
[VL/HCC'25]

aporia: decision-oriented programming with agents
[in progress]

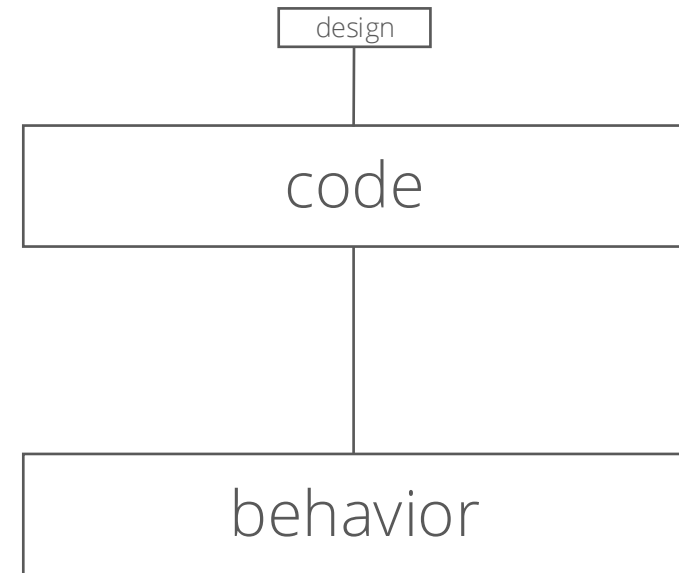
LEAP

context: code completion

programmer: validates code
by inspecting behavior



leap: uses live programming to
1) lower the cost of inspection
2) tie behavior to code



demo

user study

no-LP

AI suggestions
+
terminal

LP

AI suggestions
+
live programming

research questions

how does **live programming** affect...

1. over- / under-reliance on AI
2. validation strategies
3. cognitive load

tasks

API-heavy

pandas

clean dataframe and compute stats
using pandas

multiple correct suggestions

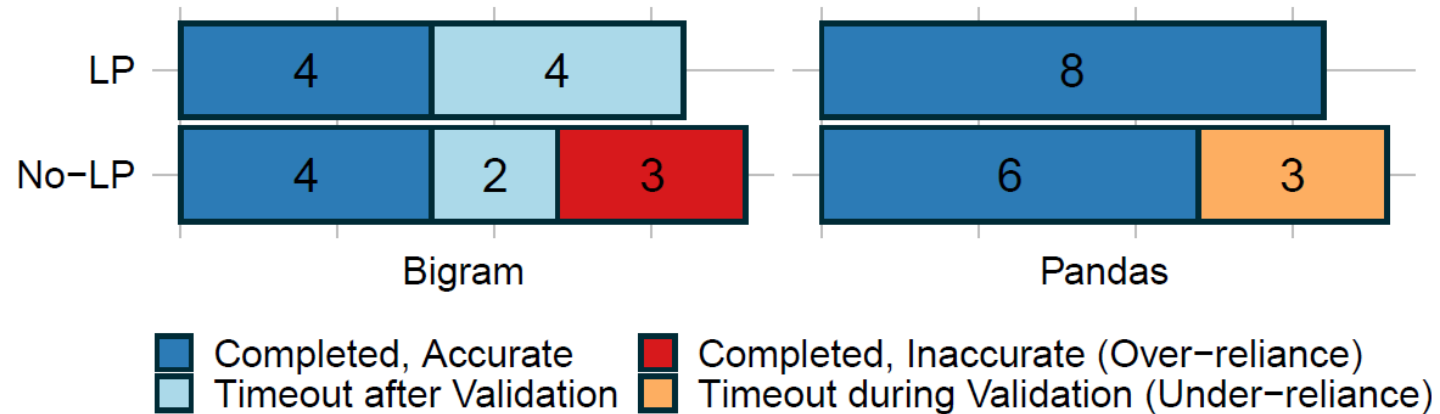
algorithmic

bigrams

find most frequent bigram in a string

no fully correct suggestions

rq1: over-/under-reliance



6 no-LP vs 0 LP participants **mid-judged** correctness of their solution

by lowering the cost of validation,
leap reduces over-/under-reliance on AI

rq1: over-/under-reliance

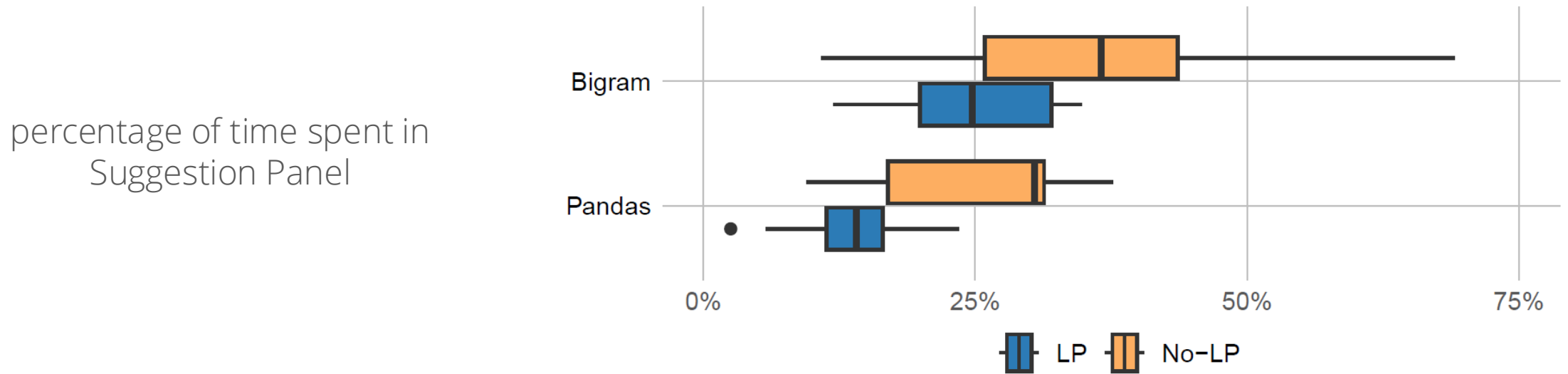
“it was **easy to understand** the behavior of a code suggestion because the little boxes on the side allowed for you to preview the results.” (P3)

“it **saved me the effort** of writing multiple print statements.” (P1)

6 no-LP vs 0 LP participants **mid-judged** correctness of their solution

by lowering the cost of validation,
leap reduces over-/under-reliance on AI

rq2: validation strategies

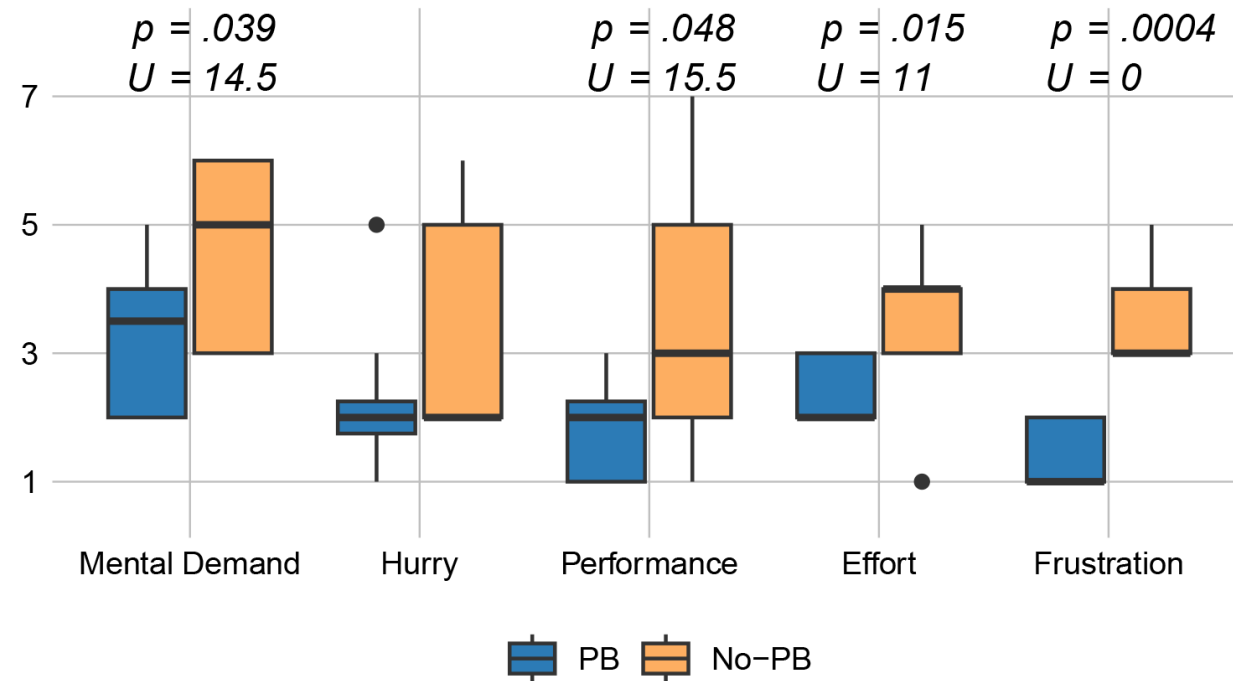


“I didn’t look too closely in the actual code,
I was *just looking at the runtime values* on the side.” (P1)

leap participants spent less time reading code

rq3: cognitive load

NASA TLX cognitive load metrics on Pandas



leap significantly reduced cognitive load of AI-assisted programming on tasks amenable to behavioral validation

this talk

leap: validating AI-generated code with live programming
[CHI'24]

› **hilde**: intentional code generation via human-in-the-loop decoding
[VL/HCC'25]

aporia: decision-oriented programming with agents
[in progress]

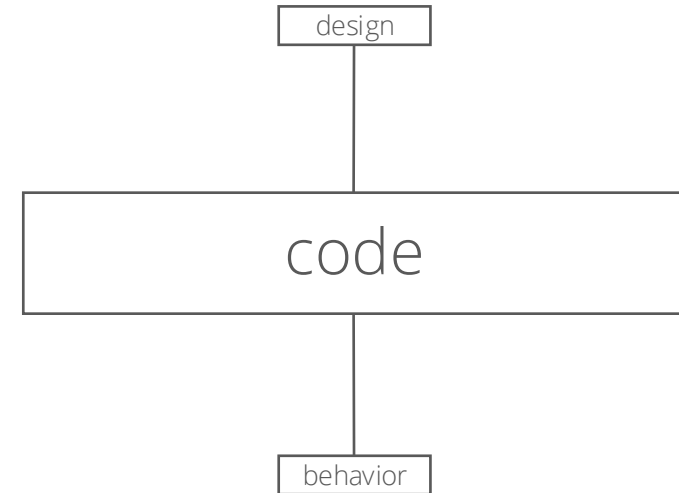
HiLDe



context: code completion



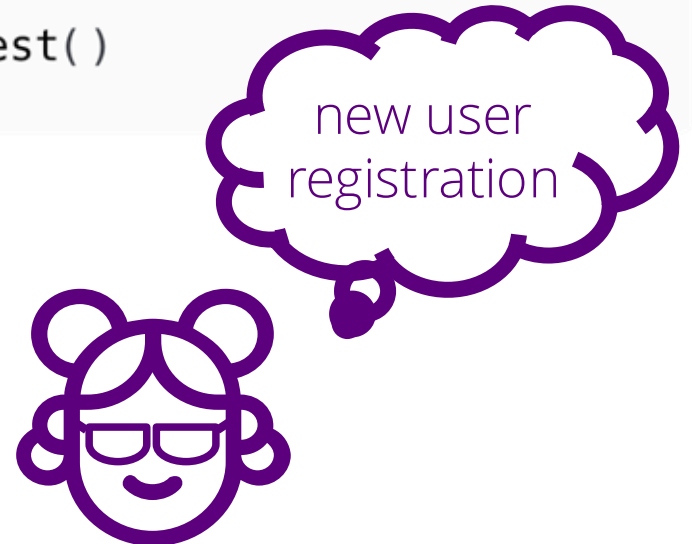
hilde: helps programmers
understand and explore the
space of implementation choices



programming in the good old days

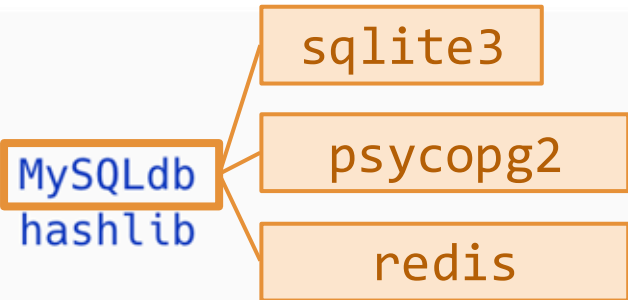


```
1 import MySQLdb
2 import hashlib
3
4 def register(username, password):
5     db = MySQLdb.connect(host="localhost", user="root", passwd="password", db="test")
6     cursor = db.cursor()
7     hashed_password = hashlib.sha256(password.encode()).hexdigest()
```

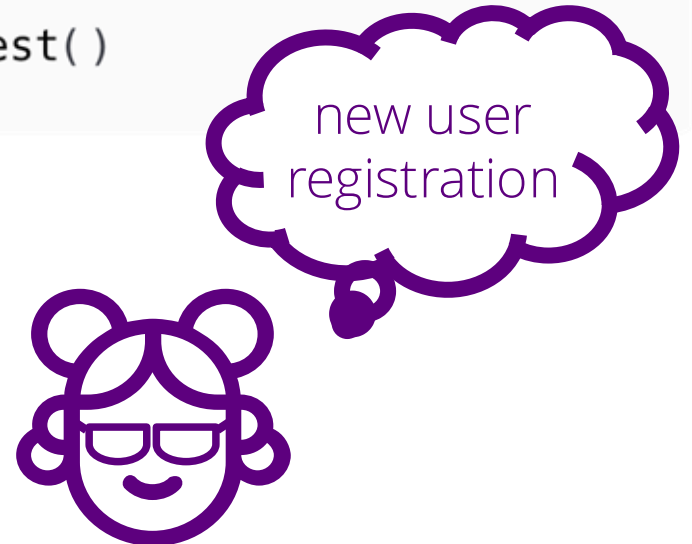


programming in the good old days

```
1 import MySQLdb
2 import hashlib
3
4 def register(username, password):
5     db = MySQLdb.connect(host="localhost", user="root", passwd="password", db="test")
6     cursor = db.cursor()
7     hashed_password = hashlib.sha256(password.encode()).hexdigest()
```



A diagram showing three orange boxes labeled 'sqlite3', 'psycopg2', and 'redis' connected by lines to a larger orange box labeled 'MySQLdb' in the code above. The 'MySQLdb' box is also highlighted with a blue border in the code.



programming in the good old days



```
1 import MySQLdb
2 import hashlib
3
4 def register(username, password):
5     db = MySQLdb.connect(host="localhost", user="root", passwd="password", db="test")
6     cursor = db.cursor()
7     hashed_password = hashlib.sha256(password.encode()).hexdigest()
```

user_role

time_stamp

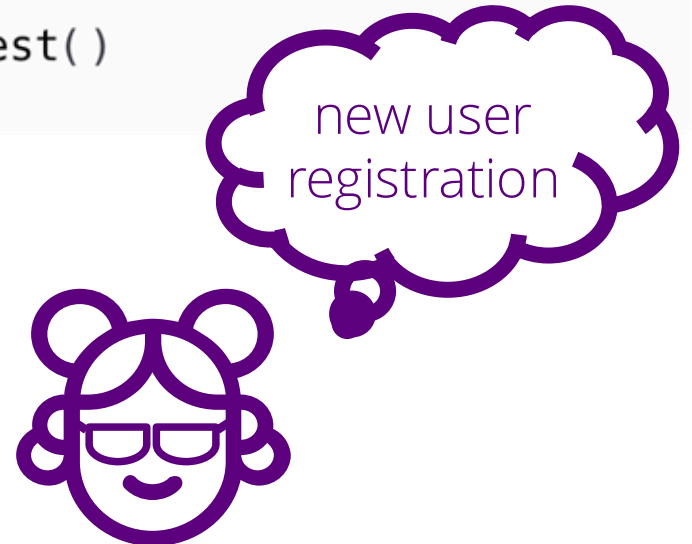
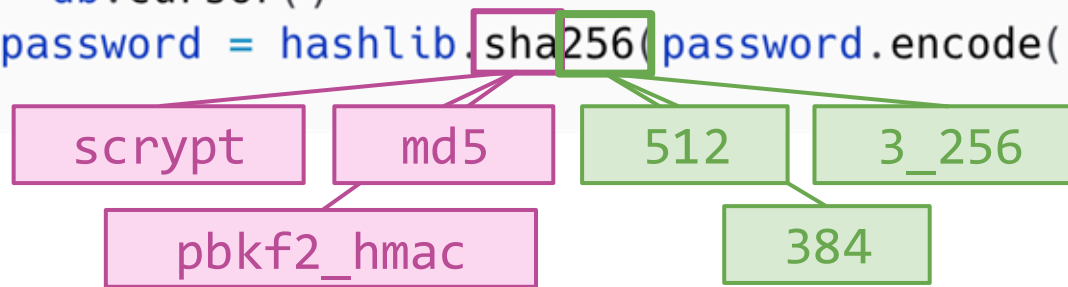
email

new user
registration



programming in the good old days

```
1 import MySQLdb
2 import hashlib
3
4 def register(username, password):
5     db = MySQLdb.connect(host="localhost", user="root", passwd="password", db="test")
6     cursor = db.cursor()
7     hashed_password = hashlib.sha256(password.encode()).hexdigest()
```



programming in the good old days

```
1 import MySQLdb
2 import hashlib
3
4 def register(username, password):
5     db = MySQLdb.connect(host="localhost", user="root", passwd="password", db="test")
6     cursor = db.cursor()
7     hashed_password = hashlib.sha256(password.encode()).hexdigest()
```

sqlite3

psycopg2

redis

MySQLdb

user_role

time_stamp

email

scrypt

md5

pbkf2_hmac

512

3_256

384

new user registration

AI obscures decision making

```
1 import MySQLdb
2 import hashlib
3
4 def register(username, password):
5     db = MySQLdb.connect(host="localhost", user="root", passwd="password", db="test")
6     cursor = db.cursor()
7     hashed_password = hashlib.sha256(password.encode()).hexdigest()
```



X or ✓ ?

AI obscures decision making

```
1 import MySQLdb
2 import hashlib
3
4 def register(username, password):
5     db = MySQLdb.connect(host="localhost", user="root", passwd="password", db="test")
6     cursor = db.cursor()
7     hashed_password = hashlib.sha256(password.encode()).hexdigest()
```

sqlite3

psycopg2

redis

user_role

time_stamp

email

scrypt

md5

512

3_256

pbkdf2_hmac

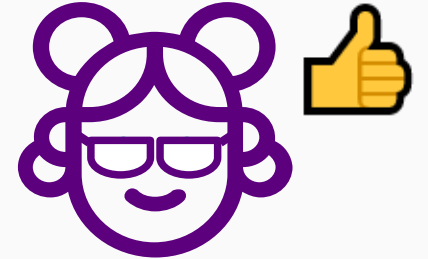
384



programmers rubber-stamp completions



```
1 import MySQLdb
2 import hashlib
3
4 def register(username, password):
5     db = MySQLdb.connect(host="localhost", user="root", passwd="password", db="test")
6     cursor = db.cursor()
7     hashed_password = hashlib.sha256(password.encode()).hexdigest()
```



X or ✓ ?

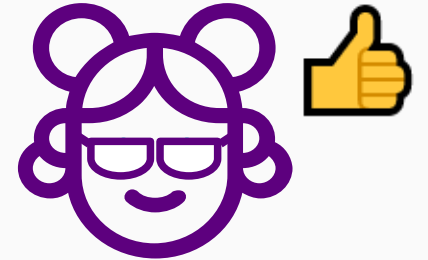
programmers rubber-stamp completions



```
1 import MySQLdb
2 import hashlib
3
4 def register(username, password):
5     db = MySQLdb.connect(host="localhost", user="root", password=password, db="test")
6     cursor = db.cursor()
7     hashed_password = hashlib.sha256(password.encode()).hexdigest()
```

lack of awareness

over-reliance



X or ✓ ?

loss of agency

real-world impact: security

2024 IEEE Symposium on Security and Privacy (SP)

Poisoned ChatGPT Finds Work for Idle Hands: Exploring Developers' Coding Practices with Insecure Suggestions from Poisoned AI Models

Sanghak Oh^{1*}, Kiho Lee^{1*}, Seonhye Park¹, Doowon Kim², Hyounghick Kim¹

¹*Department of Electrical and Computer Engineering, Sungkyunkwan University, Republic of Korea*

²*Department of Electrical Engineering and Computer Science, University of Tennessee, USA*

¹{sanghak, kiho, qkrtjsgp08, hyoung}@skku.edu, ²doowon@utk.edu

real-world impact: security

2024 IEEE Symposium on Security and Privacy (SP)

Poisoned ChatGPT Finds Work for Idle Hands: Exploring Developers' Coding Practices with Insecure Suggestions from Poisoned AI Models

Sanghak Oh^{1*}, Kiho Lee^{1*}, Seonhye Park¹, Doowon Kim², Hyounghick Kim¹

¹Department of Electrical and Computer Engineering, Sungkyunkwan University, Republic of Korea

²Department of Electrical Engineering and Computer Science, University of Tennessee, USA

¹{sanghak, kiho, qkrtjsgp08, hyoung}@skku.edu, ²doowon@utk.edu

2022 IEEE Symposium on Security and Privacy (SP)

Asleep at the Keyboard? Assessing the Security of GitHub Copilot's Code Contributions

Hammond Pearce	Baleegh Ahmad	Benjamin Tan	Brendan Dolan-Gavitt	Ramesh Karri
Department of ECE	Department of ECE	Department of ESE	Department of CSE	Department of ECE
New York University	New York University	University of Calgary	New York University	New York University
Brooklyn, NY, USA	Brooklyn, NY, USA	Calgary, Alberta, CA	Brooklyn, NY, USA	Brooklyn, NY, USA
hammond.pearce@nyu.edu	ba1283@nyu.edu	benjamin.tan1@ucalgary.ca	brendandg@nyu.edu	rkarri@nyu.edu

9833571

real-world impact: security

2024 IEEE Symposium on Security and Privacy (SP)

Poisoned ChatGPT Finds Work for Idle Hands: Exploring Developers' Coding Practices with Insecure Suggestions from Poisoned AI Models

Sanghak Oh^{1*}, Kiho Lee^{1*}, Seonhye Park¹, Doowon Kim², Hyounghick Kim¹

¹Department of Electrical and Computer Engineering, Sungkyunkwan University, Republic of Korea

²Department of Electrical Engineering and Computer Science, University of Tennessee, USA

¹{sanghak, kiho, qkrtjsgp08, hyoung}@skku.edu, ²doowon@utk.edu

2022 IEEE Symposium on Security and Privacy (SP)

Asleep at the Keyboard? Assessing the Security of GitHub Copilot's Code Contributions

Hammond Pearce Baleegh Ahmad Benjamin Tan Brendan Dolan-Gavitt Ramesh Karri
Department of ECE Department of ECE Department of ESE Department of CSE Department of ECE
New York University New York University University of Calgary New York University New York University
Brooklyn, NY, USA Brooklyn, NY, USA Calgary, Alberta, CA Brooklyn, NY, USA Brooklyn, NY, USA
hammond.pearce@nyu.edu ba1283@nyu.edu benjamin.tan1@ucalgary.ca brendandg@nyu.edu rkarri@nyu.edu

9833571



Do Users Write More Insecure Code with AI Assistants?

Neil Perry*
Stanford University

Megha Srivastava*
Stanford University

Deepak Kumar
Stanford University / UC
San Diego

Dan Boneh
Stanford University

ABSTRACT

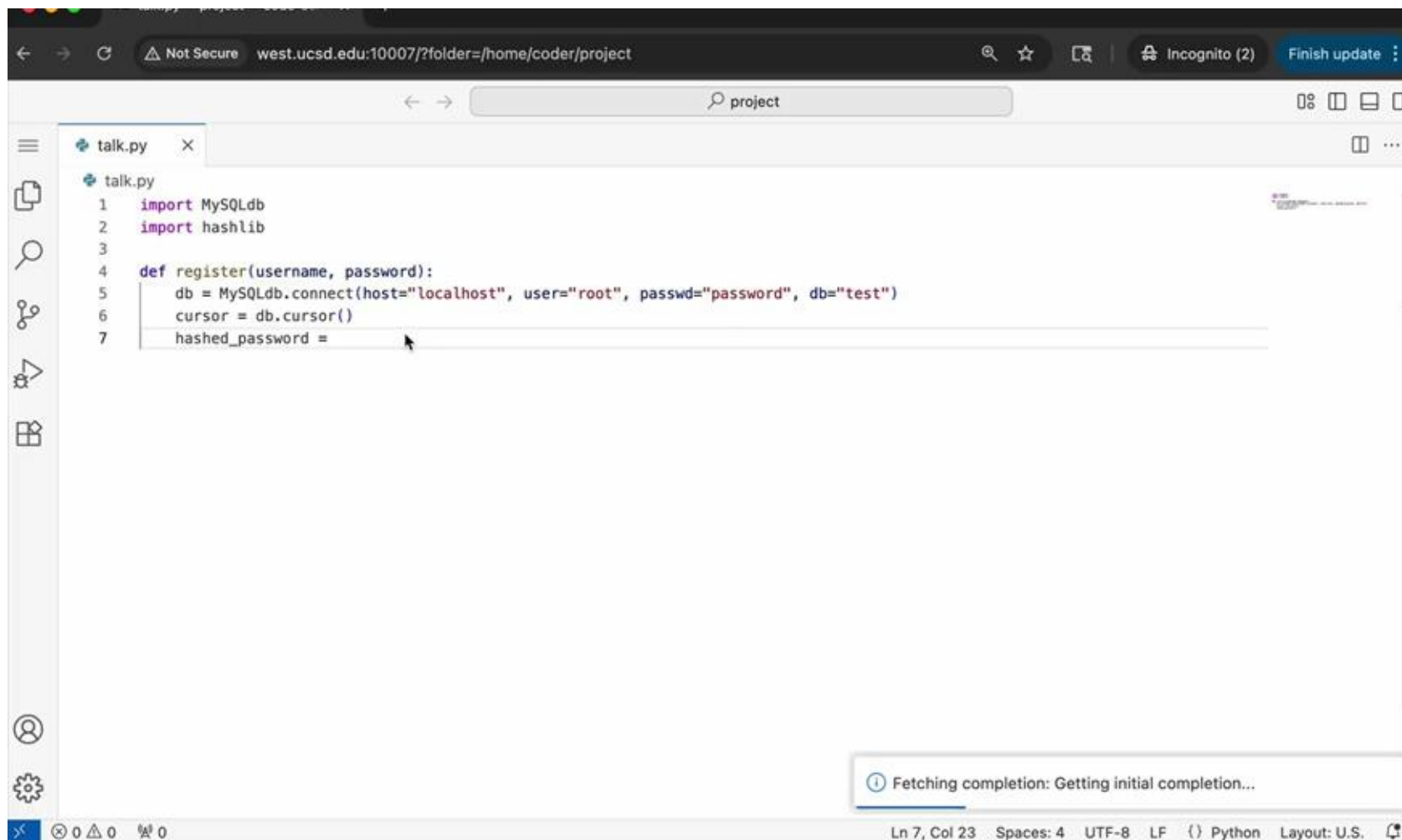
AI code assistants have emerged as powerful tools that can aid in the software development life-cycle and can improve developer productivity. Unfortunately, such assistants have also been found to produce insecure code in lab environments, raising significant

use them. Such work is important in order understand the practical security challenges introduced by AI-powered code-assistants and the ways users prompt the AI systems to inadvertently cause security mistakes.

In this paper, we examine how developers choose to interact with



bringing back human agency with HiLDe



```
talk.py
talk.py
1 import MySQLdb
2 import hashlib
3
4 def register(username, password):
5     db = MySQLdb.connect(host="localhost", user="root", passwd="password", db="test")
6     cursor = db.cursor()
7     hashed_password =
```

Ln 7, Col 23 Spaces: 4 UTF-8 LF Python Layout: U.S.

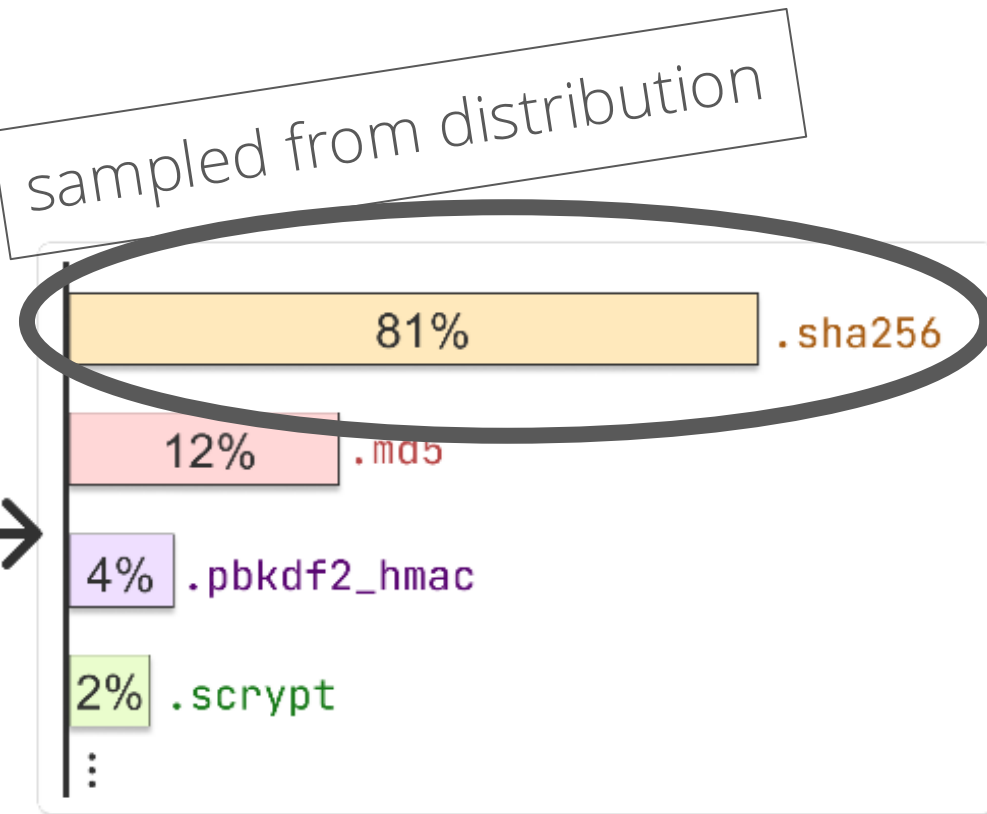
Fetching completion: Getting initial completion...

HiLDe =
Human in the Loop Decoding

decoding (traditionally)

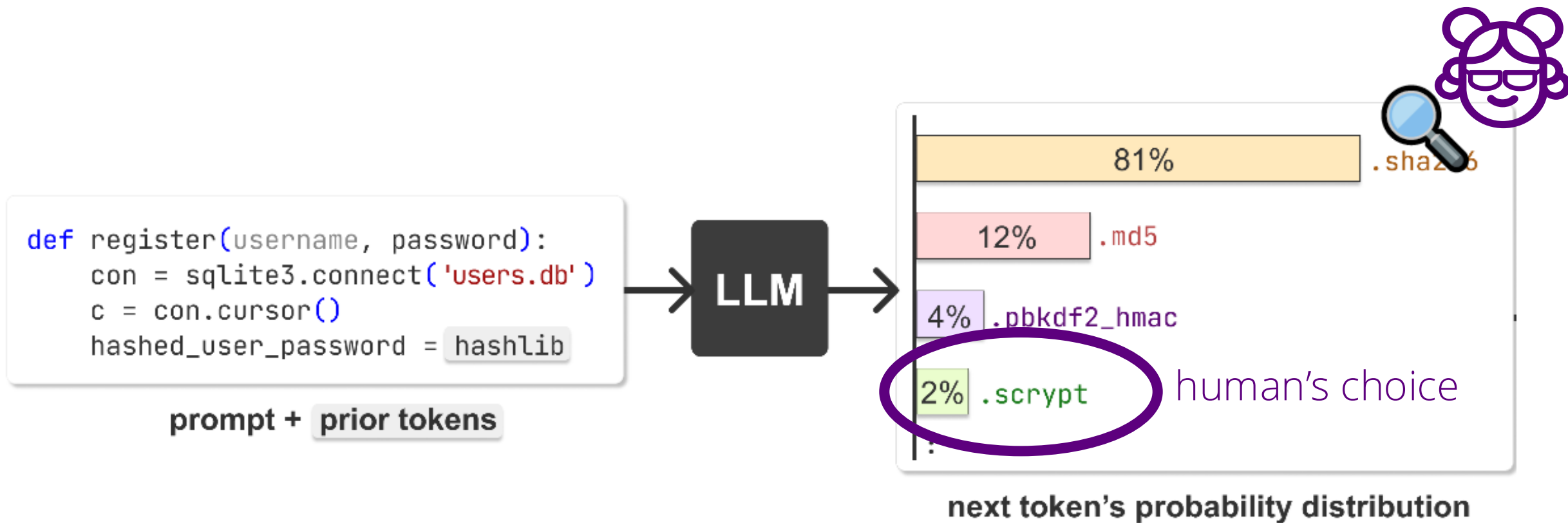
```
def register(username, password):  
    con = sqlite3.connect('users.db')  
    c = con.cursor()  
    hashed_user_password = hashlib
```

prompt + prior tokens

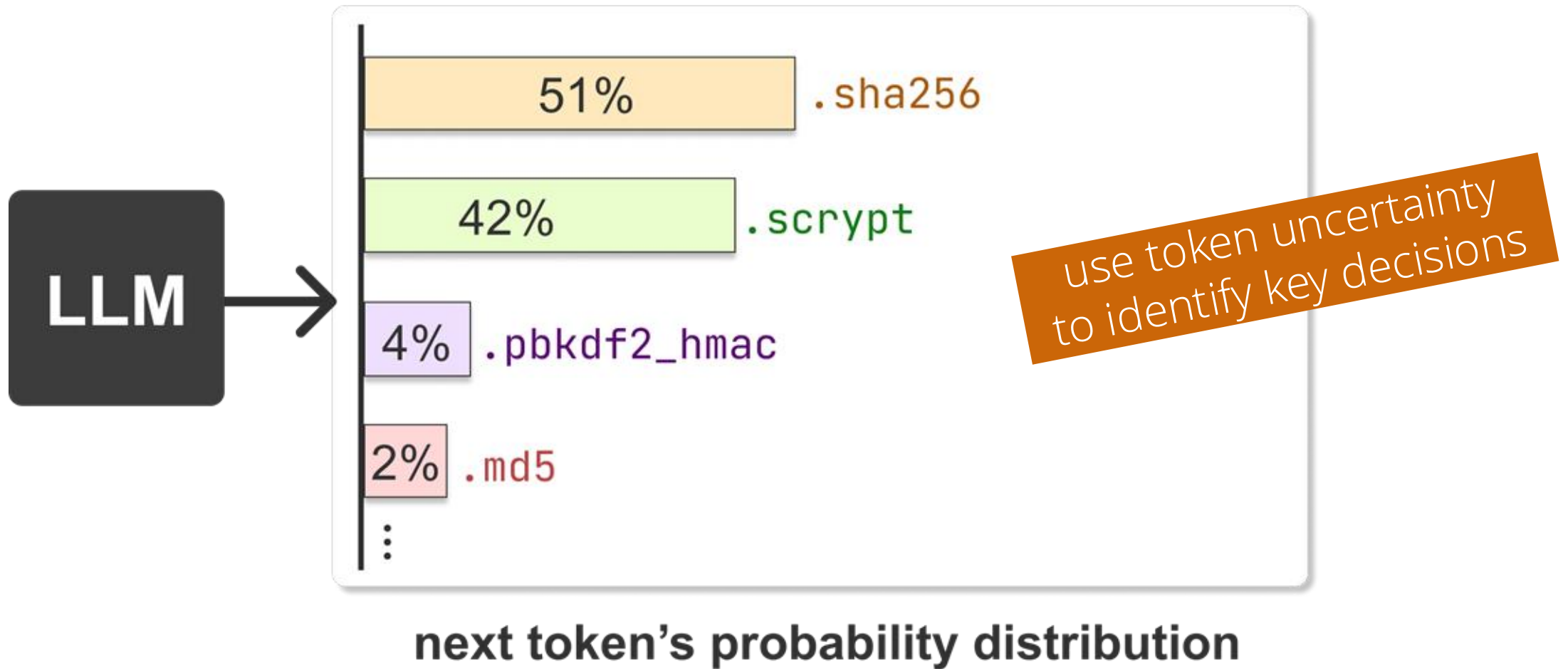


next token's probability distribution

human-in-the-loop decoding

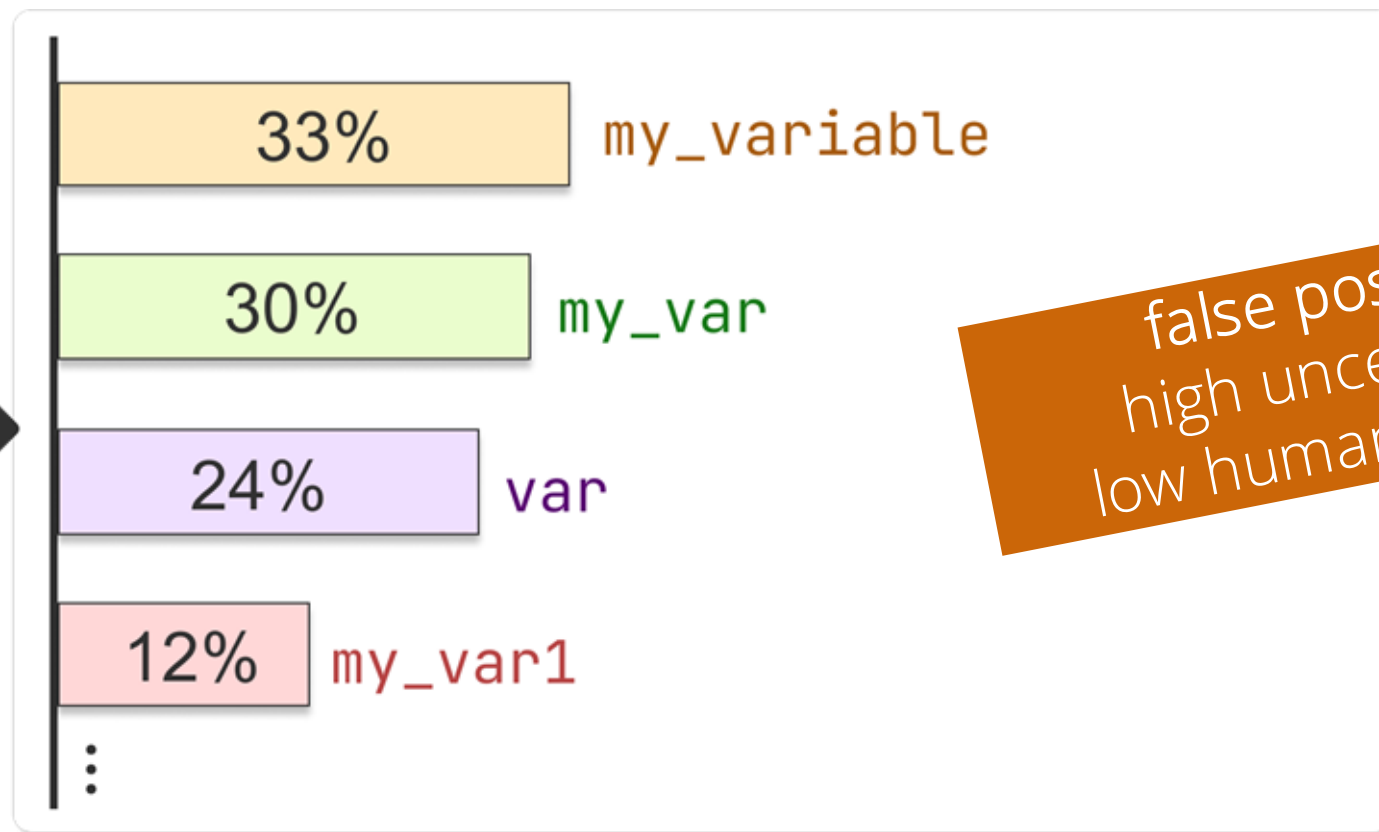


identifying key decision points



uncertainty is not enough

LLM

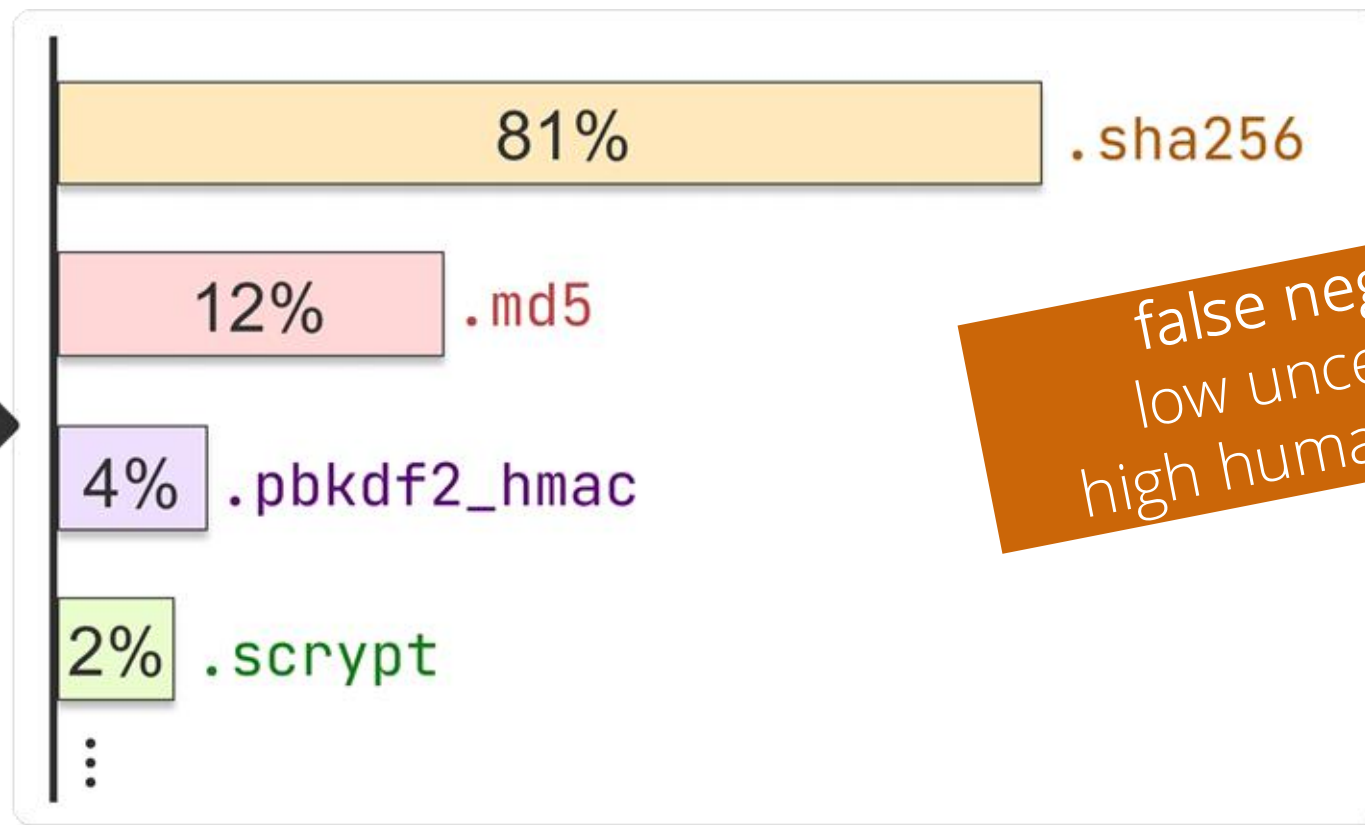


false positive:
high uncertainty
low human interest

next token's probability distribution

uncertainty is not enough

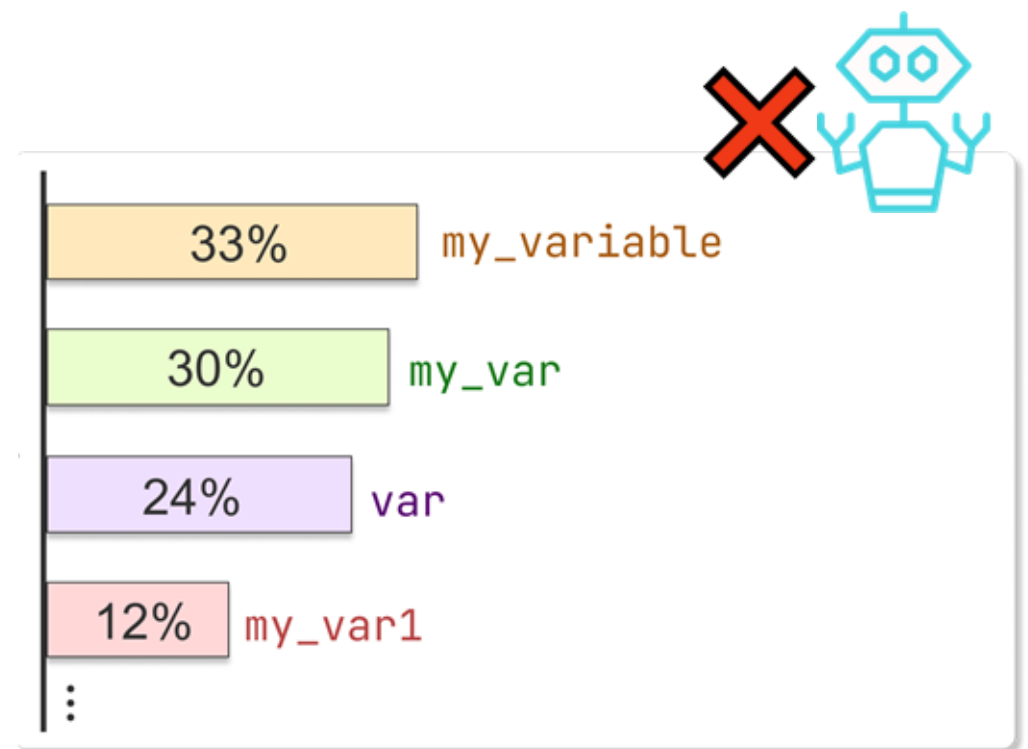
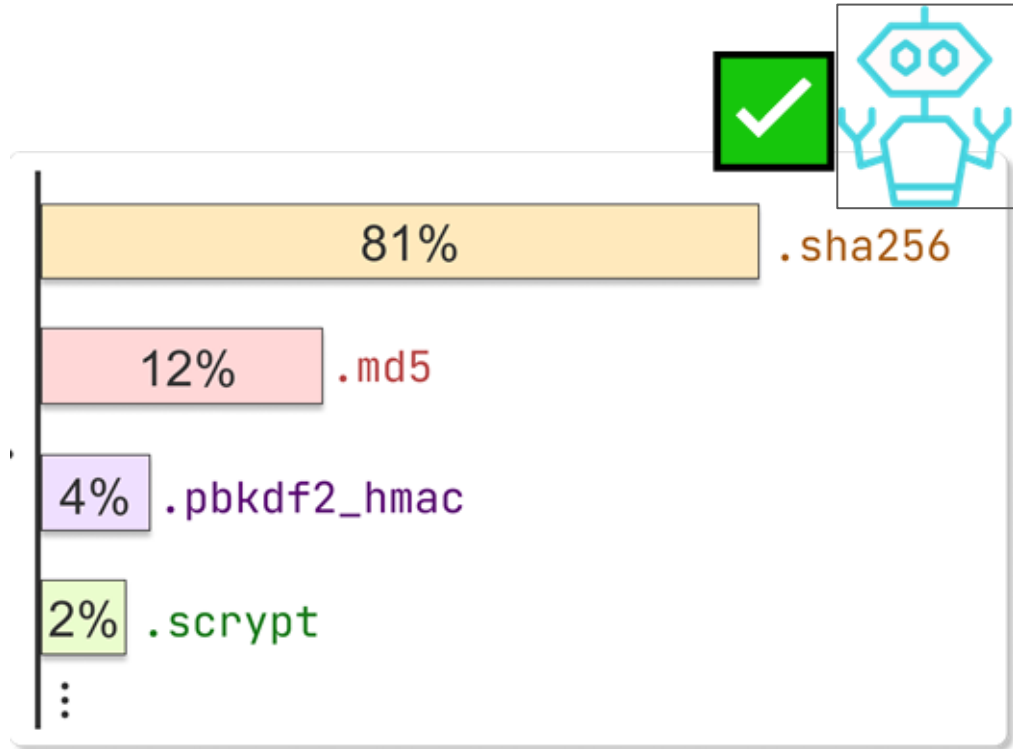
LLM



false negative:
low uncertainty
high human interest

next token's probability distribution

token uncertainty + LLM as judge



research questions

1. do users commit fewer security bugs using HiLDe?
2. how does HiLDe affect user's ability to reach their goals?

tasks

Ex. **SQL secrets** (Python):

“Given a **username** and **password**, hash the password, generate private ECDSA key, then store the username, password, and private key in a SQL database.”

procedure (n = 18)

Ex. **SQL secrets** (Python):

 15
mins

“Given a **username** and **password**, hash the password, generate private ECDSA key, then store the username, password, and private key in a SQL database.”



T1

T2

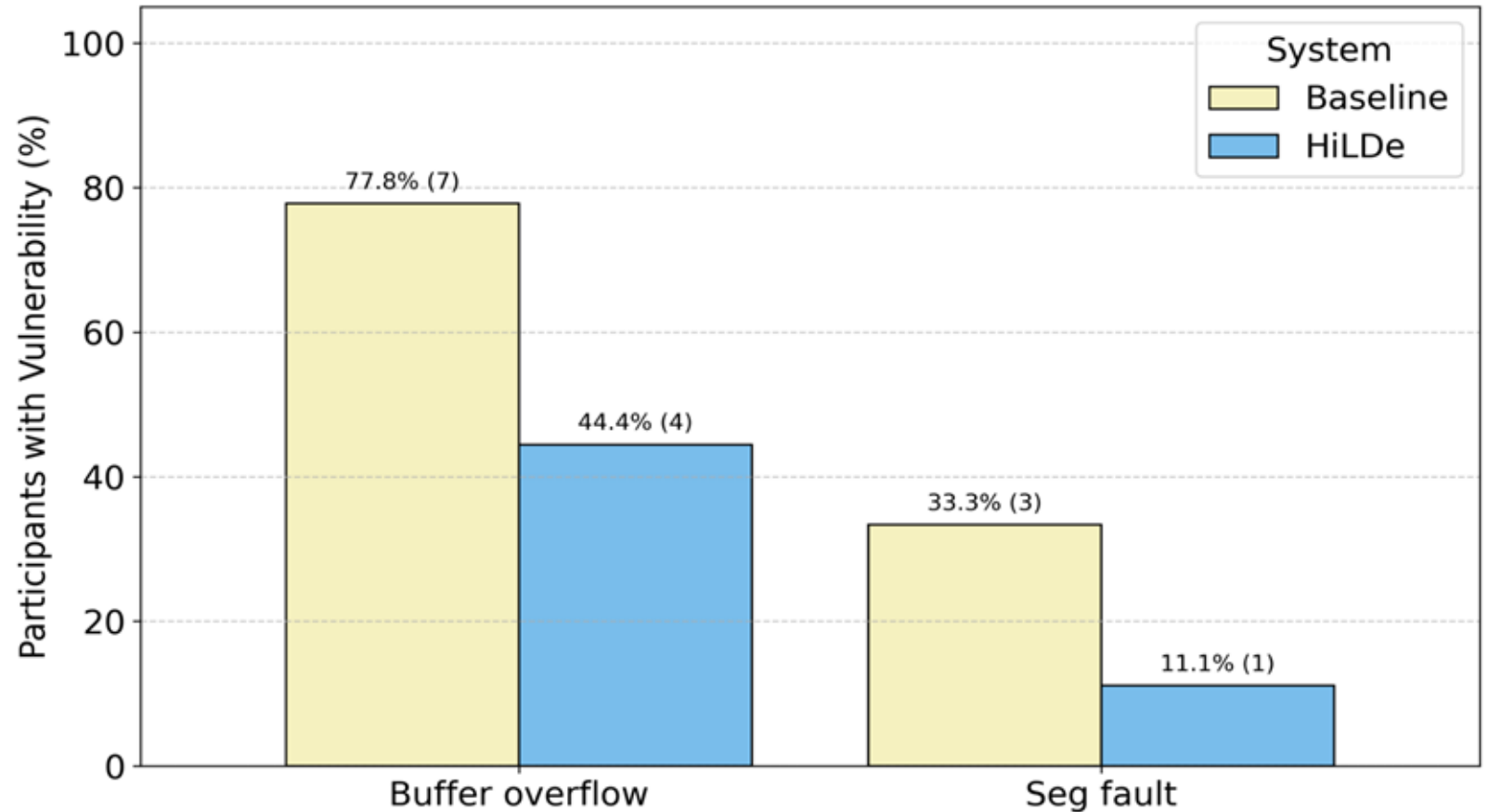


T3

T4

rq1: security

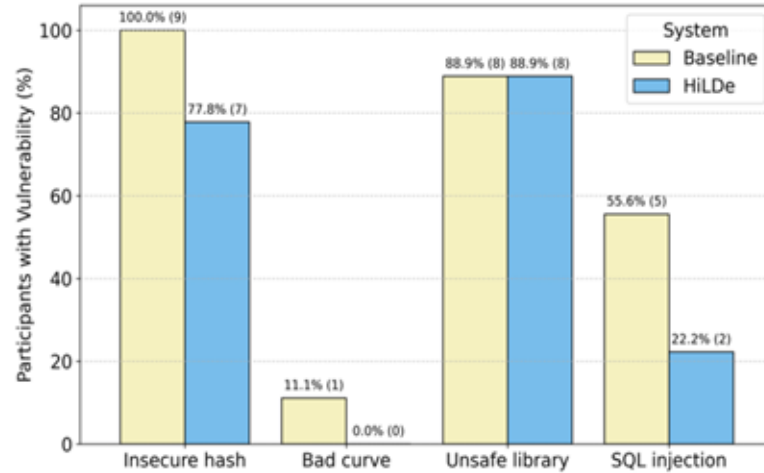
participants committed significantly fewer security bugs using HiLDe (by 30%!)



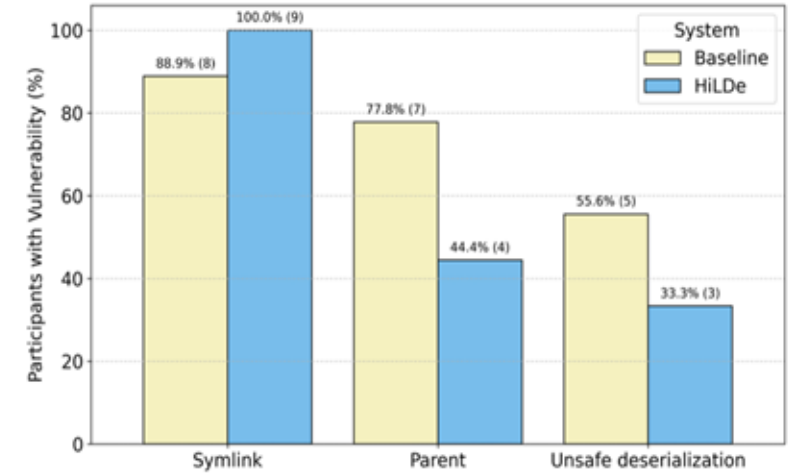
**task 4 security bugs identified in participant solutions for baseline (yellow) c*

rq1: security

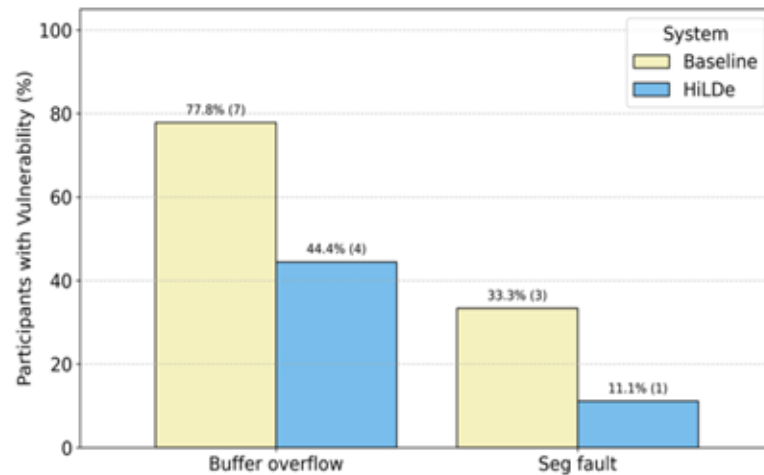
participants committed significantly fewer security bugs using HiLDe (by 30%!)



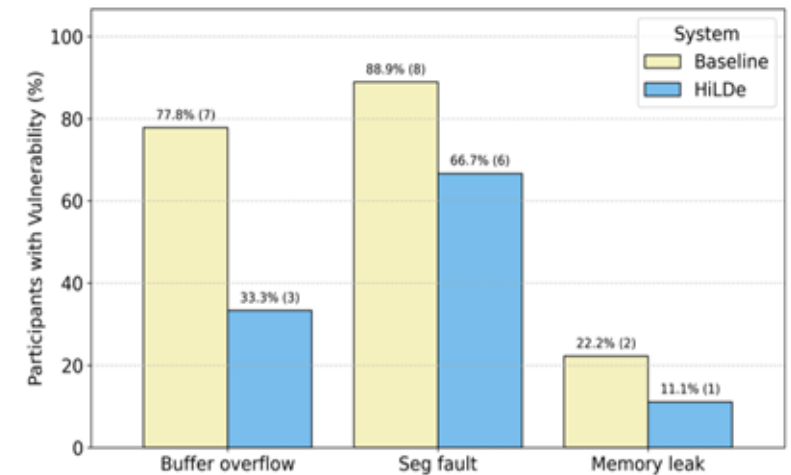
(a) SQL Secrets (T1) Mistakes



(b) Sandboxed Directory (T2) Mistakes



(c) CSV File Write (T3) Mistakes

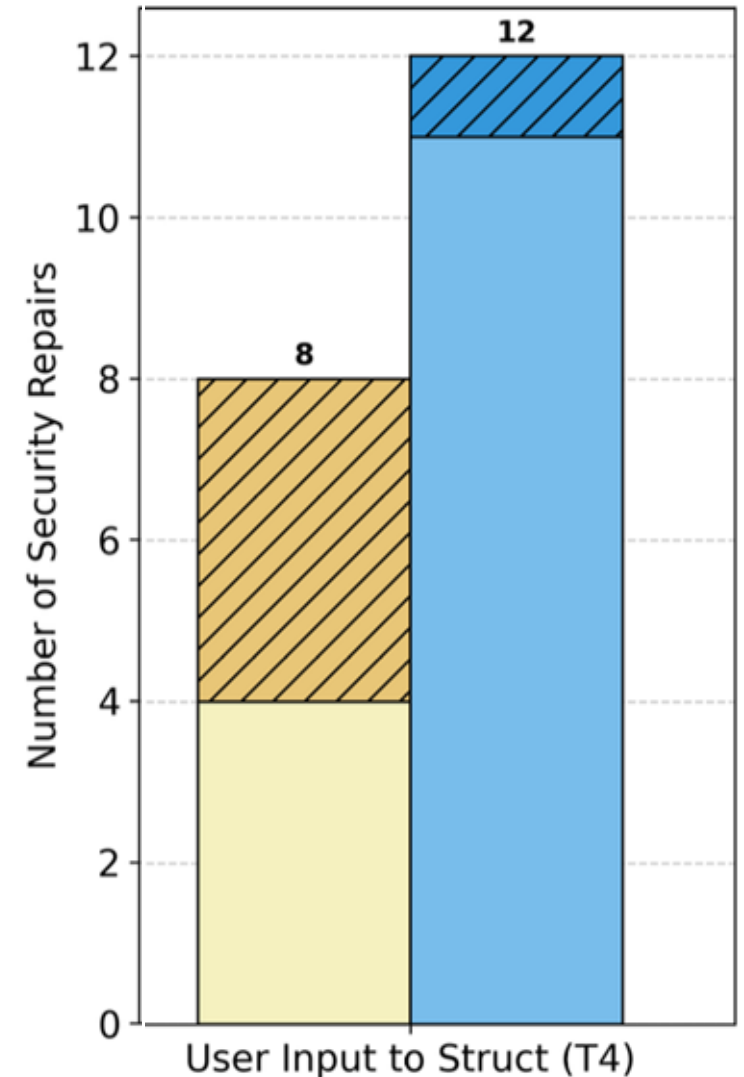
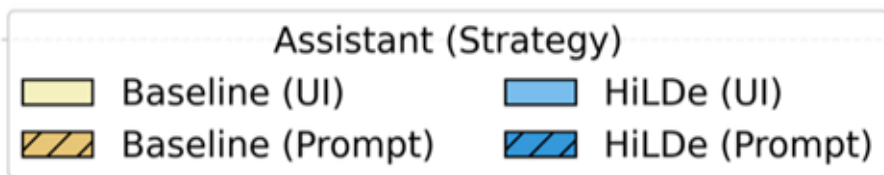


(d) User Input to Struct (T4) Mistakes

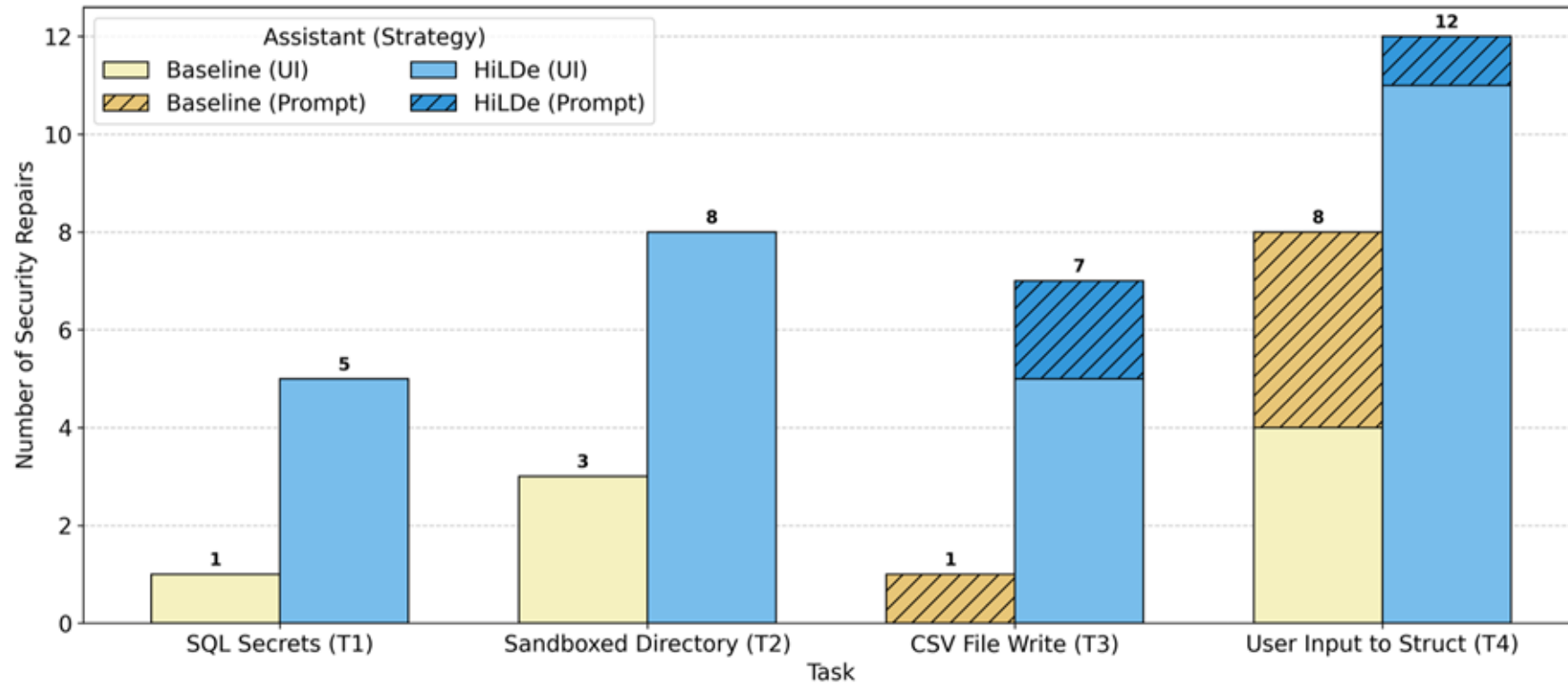
*security bugs identified in participant solutions for **baseline (yellow)** and **HiLDe (blue)**

rq1: intentional security repairs

participants **intentionally corrected** more security bugs with HiLDe



rq1: intentional security repairs



**instances of intentionally catching and fixing security bugs
an AI generated solution for **baseline (yellow)** and **HiLDe (blue)***

in

rq3: programming goals

with HilLDe, participants were able to achieve their programming goals **more effectively**

3-step interaction pattern during intentional security repair:

1. discover 
2. interpret 
3. act 

rq3: case study

1. discover 🔍
2. interpret 😲
3. act 🚗

```
int main(int argc, char *argv[]) {  
    //allocate a new person  
    < Back | > Forward | ✓ Accept (Tab) | ✕ Dismiss (Esc)  
    person *p = malloc(sizeof(person));  
    if (p == NULL) {  
        printf("Error allocating memory\n");  
        return 1;  
    }  
    printf("Enter a name: ");  
    scanf("%s", p->name);  
    p->status = 0;  
    printf("Name: %s\nStatus: %d\n", p->name, p->status);  
    free(p);  
    return 0;  
}
```

"I know with C, if you don't know what you're doing you can create some insecure stuff. So I'm assuming the LLM is just like asking if I want to use some more safe options."



P18

rq3: case study

1. discover 
2. interpret 
3. act 

Alternative Explanation: Replaces scanf with fgets to safely read a string input, preventing buffer overflows and handling spaces in input. This change improves input safety and robustness.

```
24 fgets(p->name, 100, stdin); // Uses safer input function
25 char name[100]; // Uses separate input variable
26 gets(p->name); // Uses unsafe input function
27 fscanf(stdin, "%s", p->name); // Uses fscanf for input
28 p->status = 0;
29 printf("Name: %s\nStatus: %d\n", p->name, p->status);
30 free(p);
31 return 0;
32 }
```

"fgets improves the safety by limiting the number of characters read and like yeah, buffer overflows and stuff."



P18

rq3: case study

1. discover 
2. interpret 
3. act 

```
int main(int argc, char *argv[]) {  
    //allocate a new person  
    < Back | > Forward | ✓ Accept (Tab) | × Dismiss (Esc)  
    person *p = malloc(sizeof(person));  
    if (p == NULL) {  
        printf("Error allocating memory\n");  
        return 1;  
    }  
    printf("Enter a name: ");  
    fgets(p->name, 100, stdin);  
    p->status = 0;  
    printf("Name: %sStatus: %d\n", p->name, p->status);  
    free(p);  
    return 0;  
}
```

"Yeah, let's go with this."



P18

rq3: case study

1. discover 
2. interpret 
3. act 



*"[HiLDe] wasn't sure of which function to use for user input and that's an **important decision**. It did make me **aware** of, yeah, we don't want to just accept arbitrary input."*

*"I don't really think about the security of my code. But **looking at the options**, I was sort of **motivated** to pick something that would be more secure."*



"[HiLDe] helped me better realize my intent instead of express my intent. It helped me realize my intent because I wasn't aware of it before."

rq3: case study (baseline)

"I would say the only potential problem is I think I've heard that `scanf` is not secure or something, I don't know, maybe. But I guess for the purposes of this task, it didn't really matter"



P13

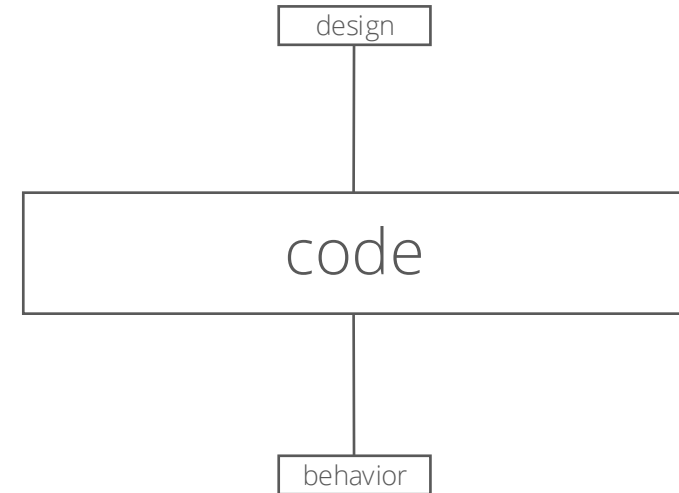
HiLDe



context: code completion



hilde: helps programmers
understand and explore the
space of implementation choices



this talk

leap: validating AI-generated code with live programming
[CHI'24]

hilde: intentional code generation via human-in-the-loop decoding
[VL/HCC'25]

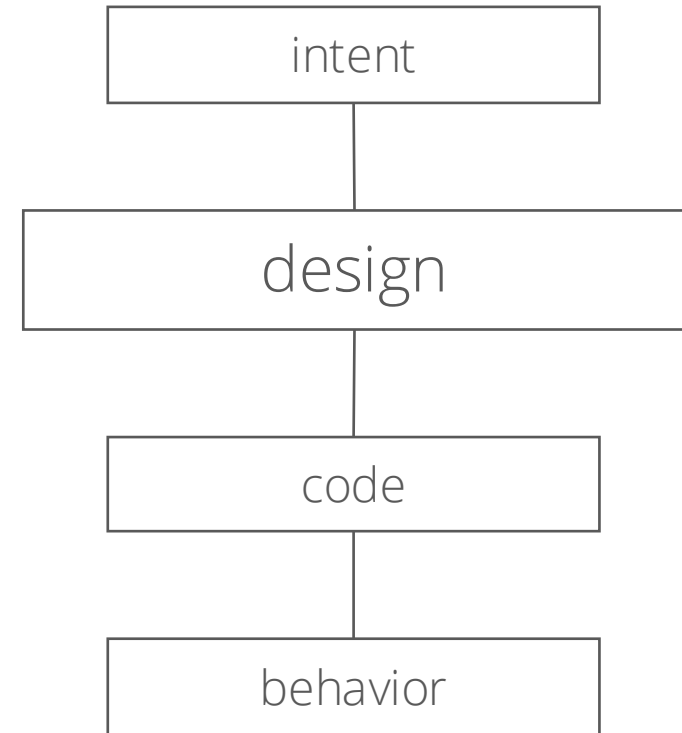
➤ **aporia**: decision-oriented programming with agents
[in progress]

aporia

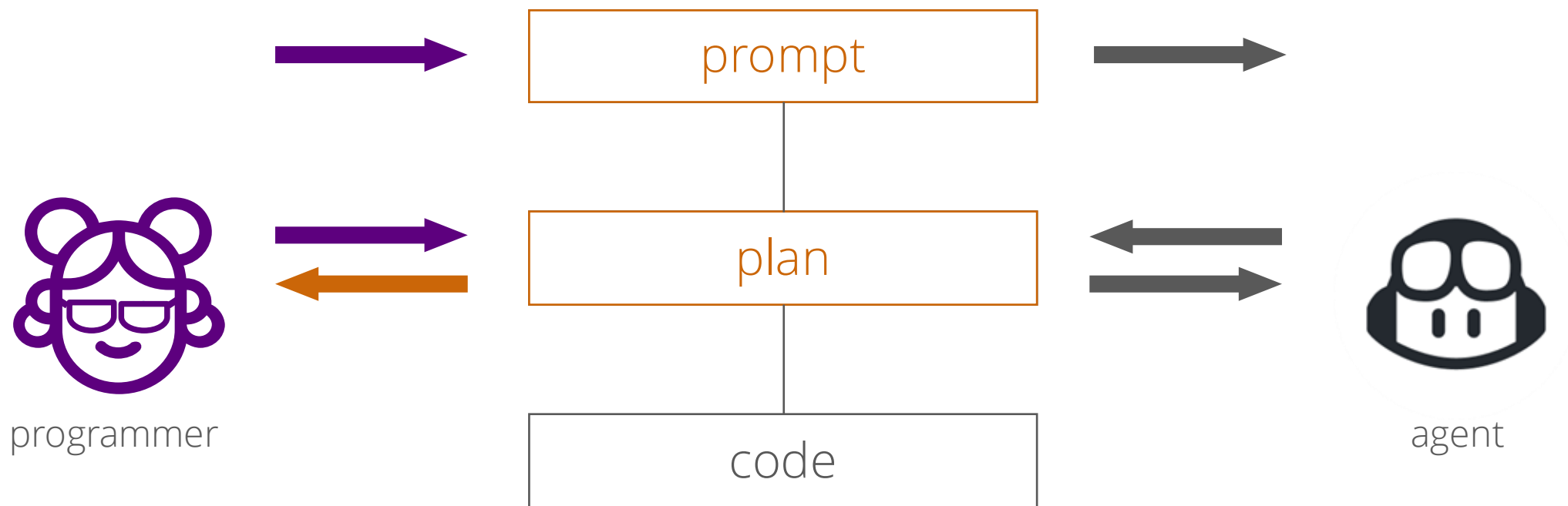
context: programming agents



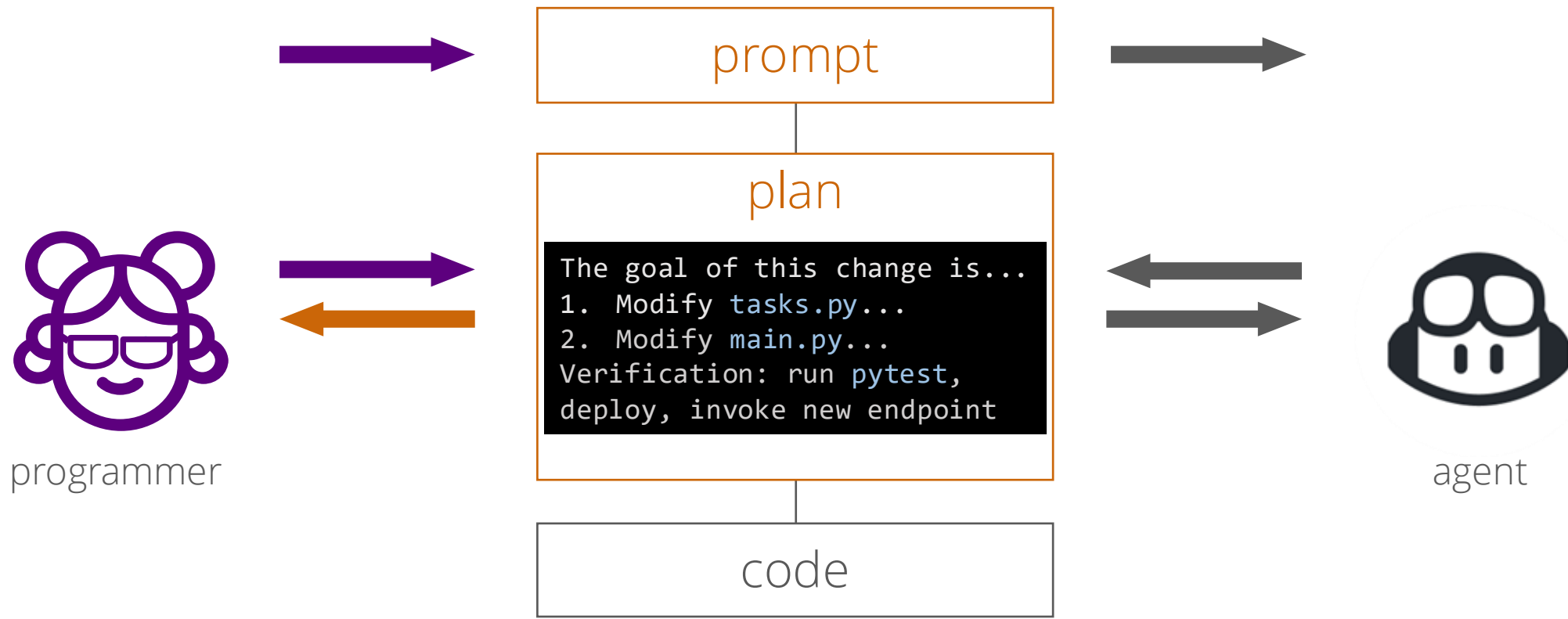
aporia: helps programmers
explore the design space



SotA: **plan-oriented** programming



SotA: **plan-oriented** programming



SotA: **plan-oriented** programming

lack of awareness



over-reliance



programmer

loss of agency

prompt

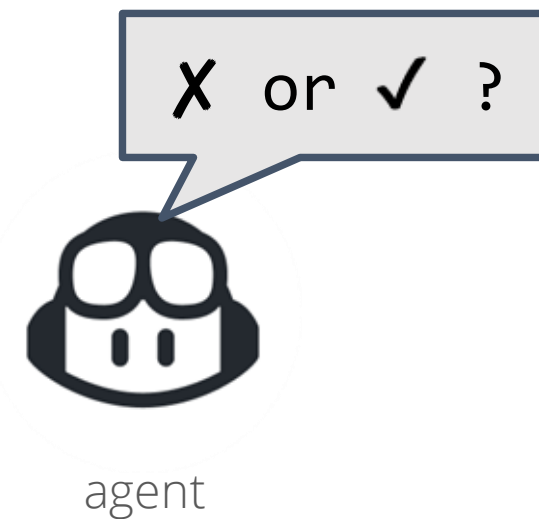


plan

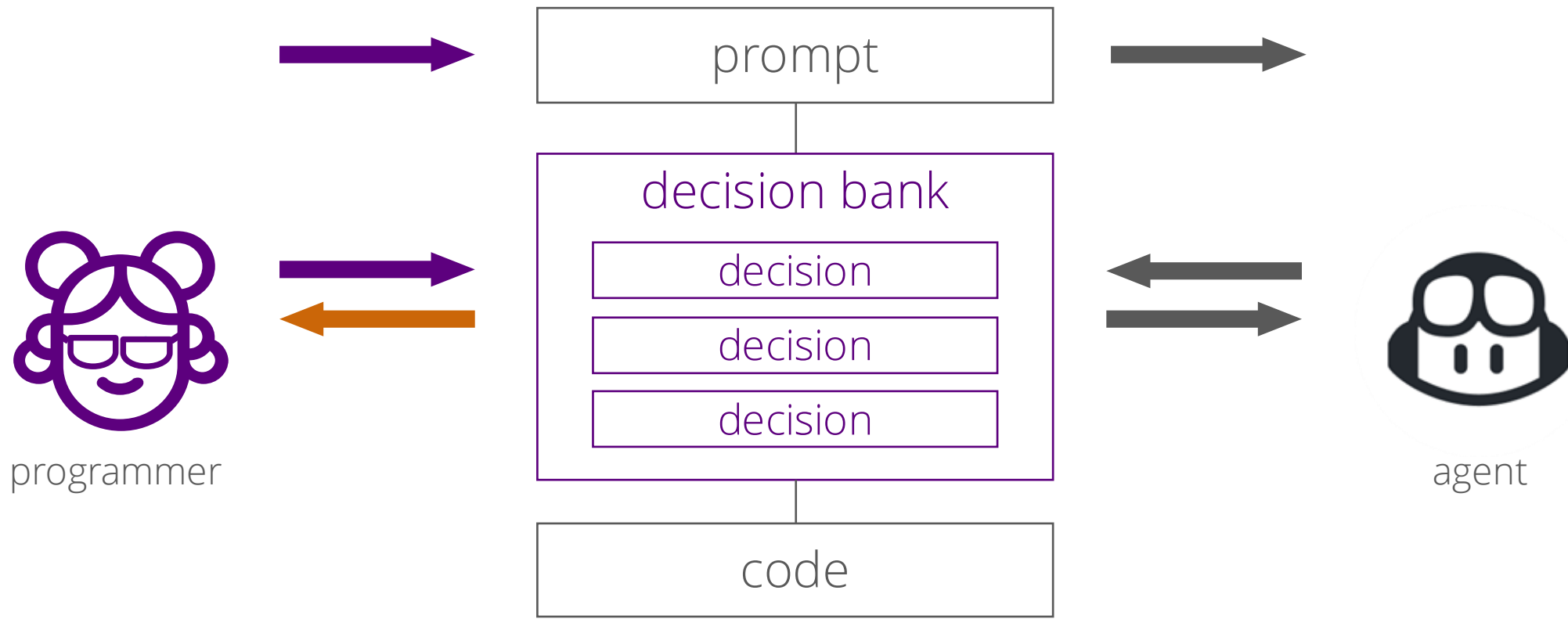
```
The goal of this change is...  
1. Modify tasks.py...  
2. Modify main.py...  
Verification: run pytest,  
deploy, invoke new endpoint
```



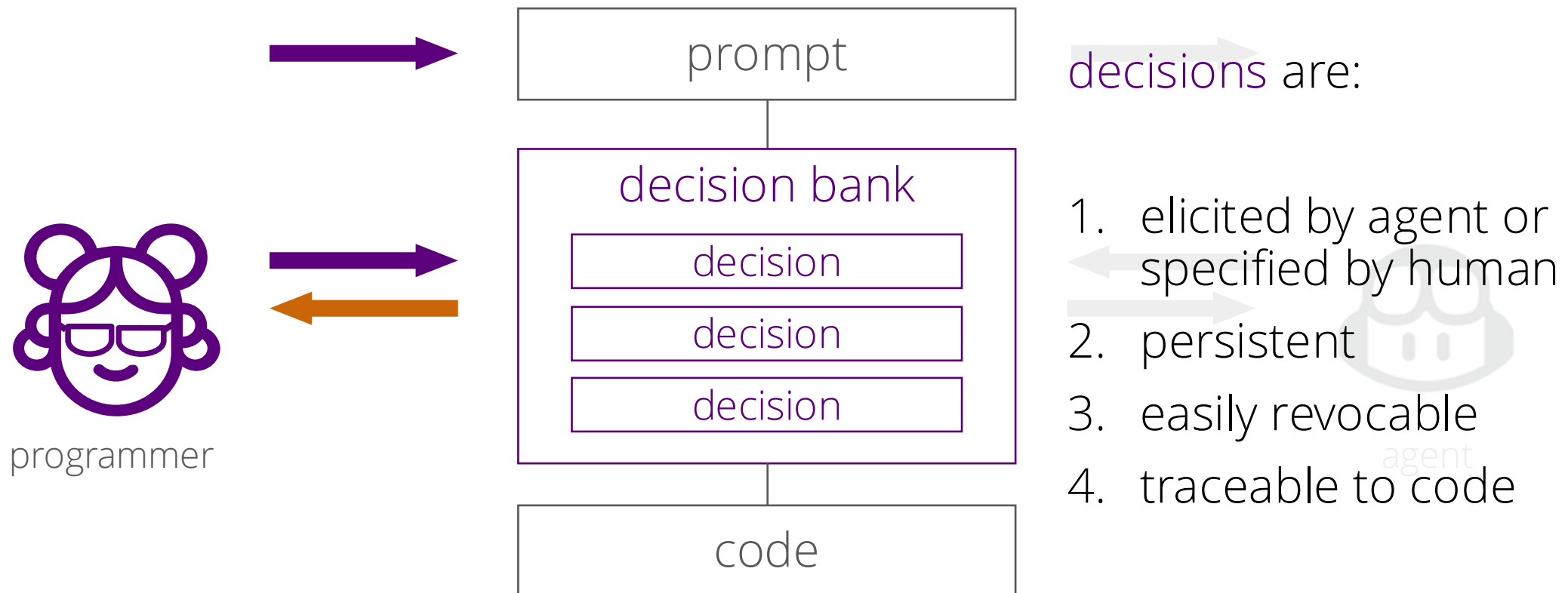
code



our idea: **decision-oriented** programming



our idea: **decision-oriented** programming



fish

test_app.py test_plan.py U X

tests > test_plan.py

```
1
```

▼ GOAL

Recommend papers for desk rejection based on reasonable criteria

RESET PLAN UPDATE GOAL

APORIA

▼ DECISIONS

▼ GOAL

Recommend papers for desk rejection based on reasonable criteria

RESET PLAN UPDATE GOAL

Add a new decision... Submit

> IMPLEMENTATION PROGRESS

> REVIEW




Ln 9, Col 29 Spaces: 4 UTF-8 LF {} Python Finish Setup Prettier

fish

test_app.py test_plan.py U X

tests > test_plan.py

1

- + Should desk rejection recommendations only include SUBMITTED status? 
- + Should papers with abstracts below minimum length be recommended for desk rejection? 
- + Should desk rejection recommendations be accessible only to admin users? 

APORIA

DECISIONS

GOAL

Recommend papers for desk rejection based on reasonable criteria

RESET PLAN UPDATE GOAL

Add a new decision... Submit

IMPLEMENTATION PROGRESS

REVIEW



Ln 9, Col 29 Spaces: 4 UTF-8 LF {} Python Finish Setup Prettier

fish

test_app.py test_plan.py U x

tests > test_plan.py

1

- + Should papers with abstracts below minimum length be recommended for desk rejection? 
- + Should desk rejection recommendations be accessible only to admin users? 

APORIA

DECISIONS

GOAL

Recommend papers for desk rejection based on reasonable criteria

RESET PLAN UPDATE GOAL

Desk rejection recommendations should only include papers in SUBMITTED status.

Add a new decision... Submit

IMPLEMENTATION PROGRESS

REVIEW

Ln 9, Col 29 Spaces: 4 UTF-8 LF {} Python Finish Setup Prettier

fish

test_app.py test_plan.py U x

tests > test_plan.py

```
1
+ Should desk rejection recommendations be accessible only to admin users?
```

APORIA

DECISIONS

GOAL

Recommend papers for desk rejection based on reasonable criteria

RESET PLAN UPDATE GOAL

Desk rejection recommendations should only include papers in SUBMITTED status.

Papers with abstracts below minimum length should be recommended for desk rejection.

Add a new decision... Submit

IMPLEMENTATION PROGRESS

REVIEW

master* Launchpad 0 0 Ln 9, Col 29 Spaces: 4 UTF-8 LF {} Python Finish Setup Prettier

fish

test_app.py test_plan.py U x

tests > test_plan.py

```
1
```

APORIA

DECISIONS

GOAL

Recommend papers for desk rejection based on reasonable criteria

RESET PLAN UPDATE GOAL

Desk rejection recommendations should only include papers in SUBMITTED status.

Papers with abstracts below minimum length should be recommended for desk rejection.

Desk rejection recommendations should be accessible only to admin users.

Add a new decision... Submit

IMPLEMENTATION PROGRESS

REVIEW

master* Launchpad 0 0 Ln 9, Col 29 Spaces: 4 UTF-8 LF {} Python Finish Setup Prettier

fish

test_app.py test_plan.py U X

```
tests > test_plan.py
1  implement plan | regenerate plan | ? regenerate questions
   # test_plan.py
   + Should papers with titles below a minimum length be recommended for desk rejection? X
   + Should the desk rejection recommendations page render flagged paper details in the HTML response? X
   + Should a paper meeting multiple rejection criteria appear exactly once in recommendations? X
   + Should papers with no attached PDF be recommended for desk rejection? X
   + Should `get_desk_rejection_recommendations` return an empty list when given an empty input? X
   + Should all flagged papers appear in recommendations when the input contains both flagged and unflagged papers? X

2
3  from datetime import datetime
4
5  from notcrp.models import Paper, PaperStatus, User, UserRole
6  from notcrp.routes.dashboard import get_desk_rejection_recommendations
7
8  LONG_ABSTRACT = "A" * 200
9  SHORT_ABSTRACT = "Too vague"
10
11
12 > class TestSubmittedOnlyRecommendations: --
   ✓ Should desk rejection recommendations only include papers in SUBMITTED status?
56
57
58 > class TestShortAbstractRecommendation: --
   ✓ Should papers with abstracts below a minimum length be recommended for desk rejection?
88
89
90 > class TestDeskRejectionAdminOnly: --
   ✓ Should desk rejection recommendations be accessible only to admin users?
126
```

decisions turn into tests

APORIA

DECISIONS

GOAL

Recommend papers for desk rejection based on reasonable criteria

RESET PLAN UPDATE GOAL

ALL YES NO DISMISS

Should papers with very short titles be recommended for desk rejection?

Papers with abstracts below minimum length should be recommended for desk rejection.

Desk rejection recommendations should only include papers in SUBMITTED status.

Should papers with titles below a minimum length be recommended for desk rejection?

Desk rejection recommendations should be accessible only to admin users.

Should desk rejection recommendations exclude papers that already have reviewer assignments?

Should papers with very short titles be recommended for desk rejection?

Could desk rejection update a paper's status without requiring all 3 reviews to be submitted?

Should a paper failing multiple criteria appear exactly once in desk rejection recommendations?

Should papers with very short titles be recommended for desk rejection?

Should each desk rejection recommendation include the specific criteria that triggered the flag?

Add a new decision... Submit

IMPLEMENTATION PROGRESS

REVIEW

Ln 9, Col 29 Spaces: 4 UTF-8 LF {} Python Finish Setup Prettier

research questions

1. does Aporia help programmers articulate intent?
2. does Aporia help programmers align code with intent?
3. does Aporia promote appropriate trust in the code?

co-active ai-assisted programming

aporia: decision-oriented programming with agents
[in progress]



hilde: intentional code generation via human-in-the-loop decoding
[VL/HCC'25]



leap: validating AI-generated code with live programming
[CHI'24]



design

code

behavior

thanks to students & collaborators!



Kasra Ferdowsi



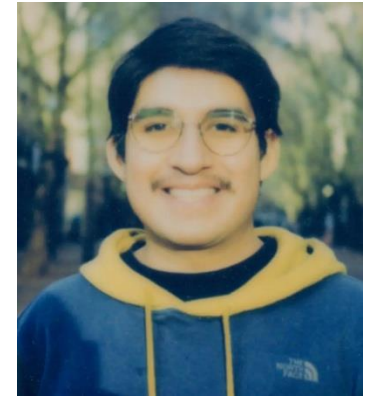
Lisa Huang



Raven Rothkopf



Saketh Kasibatla



Emmanuel Anaya
Gonzalez



Hila Peleg



Sorin Lerner



Harrison
Goldstein