Identifier Readability & Program Comprehension

To evaluate and formalize how identifier naming influences code comprehension and propose a practical model to assess readability

Presented by

Bharat B Mane

Registration No: AU/PhD/20/0308

Batch: 2020

Guided by

Dr. Rathnakar Achary

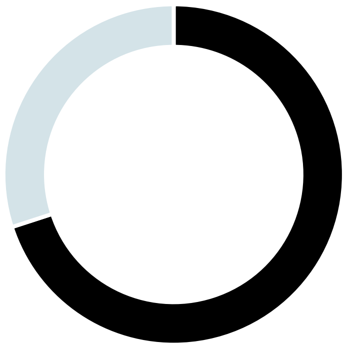
Department of Computer Science and Engineering

Alliance University

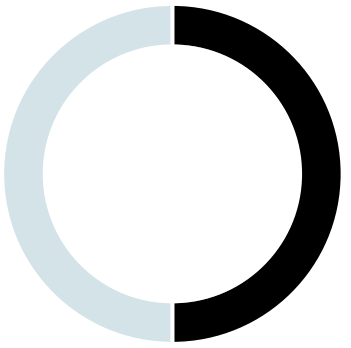
Date: 30 June 2025

The Human Cost of Unreadable Code

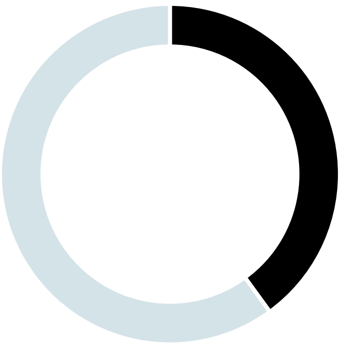
We Read Code Far More Than We Write It — So Why Don't We Name Things Better?



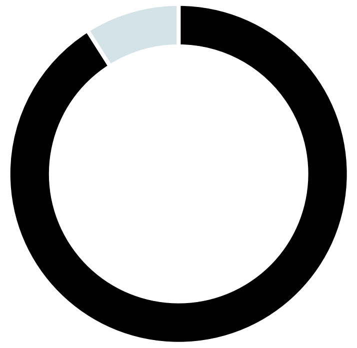
70%
Time Spent
Reading Code



50%
Comprehension
from Naming



40%
Cost Saved by
Readability



91%
Renames to
Improve Clarity

> “Code is read more often than it is written.”
> — Robert C. Martin (Clean Code)

```
int x = 0;        t
t = x * y;     f(a,b);
f(a,b);   val1 ǂ t;
val1 = t;  if (p)
f(p) returnupdate(t);
update(t);

a=/d      (rat n;
{   udate();
}
```

```
int totalRevenue = 0;
totalRevenue = price *
calculateDiscount(rate,
totalRevenue);
if (valid) return;

computeAverage();
{
  totalRevenue= = 0;
  totalRevenue= price *
  calculateDiscount(rate;
  discountRate += totarven;
```

What the Research Says: 20 Years of Evidence

2004-2024

Reviewed 52 peer-reviewed studies

SOURCES

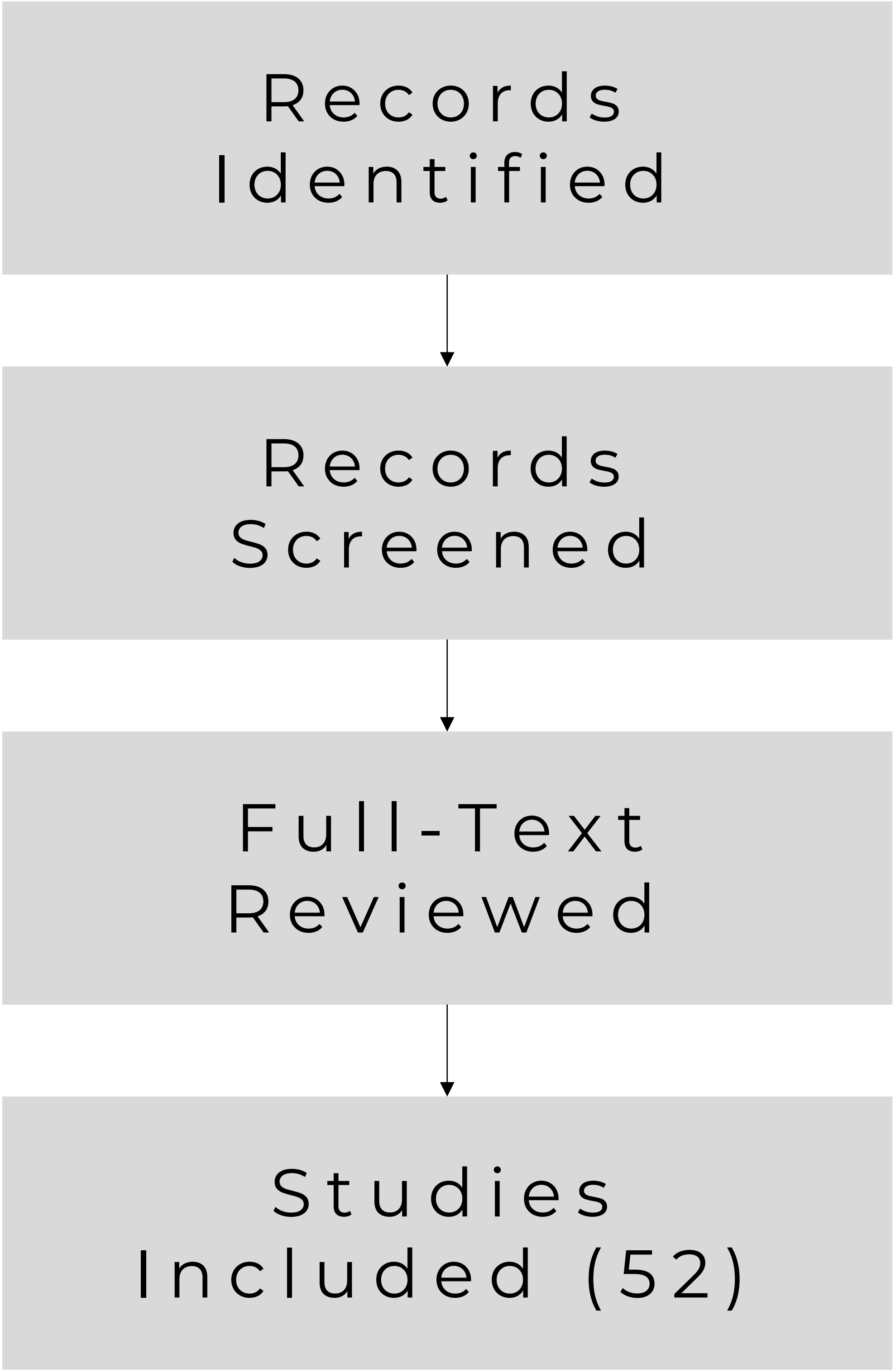
IEEE, ACM, Scopus, SpringerLink, Google Scholar.

FOCUS AREA

- Identifier Readability
- Naming Conventions
- Code Comprehension

METHOD

PRISMA (Systematic Selection & Screening)

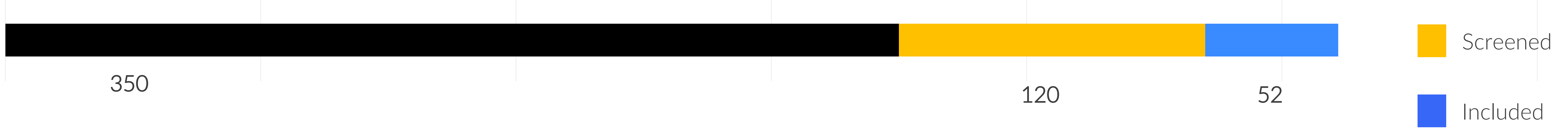


Records Identified

Records Screened

Full-Text Reviewed

Studies Included (52)



Insight: Consistent, meaningful naming improves comprehension, reduces bugs, and aids maintainability.

Why Naming Matters: What the Research Shows

Cognitive Perspective

Identifiers act as “cognitive beacons”

Recognizable names trigger mental models (Brooks, 1983)

Clear naming reduces cognitive load

Enables chunking & faster recall (Miller’s 7±2 rule)

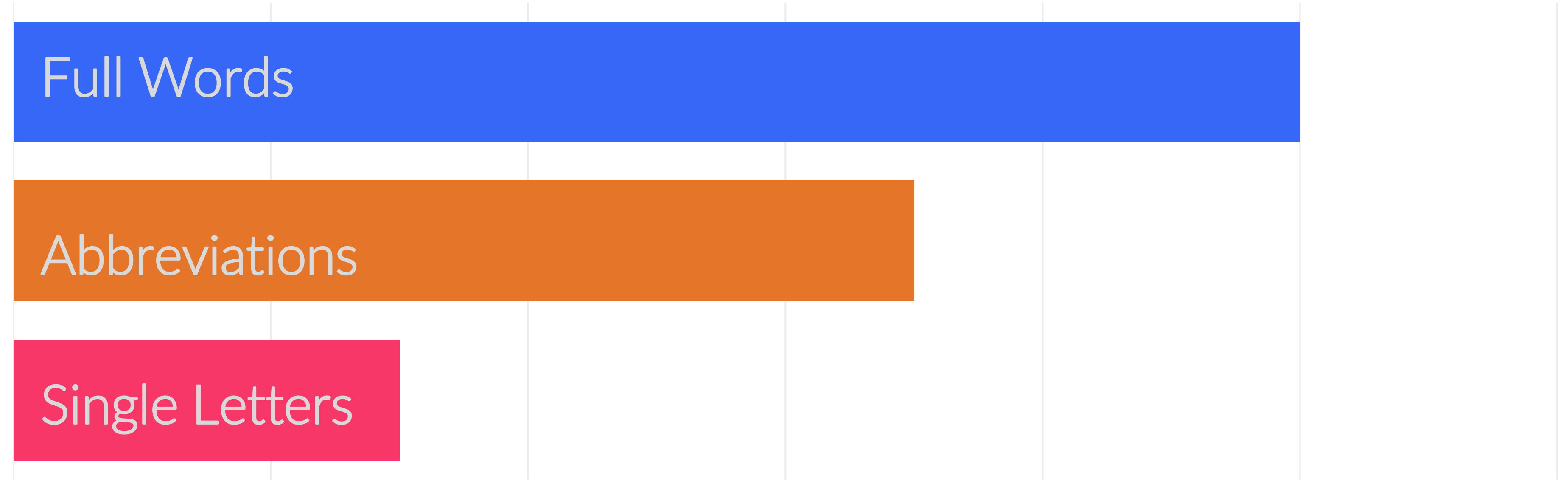
Naming Trade-offs & Style

Semantic Clarity > Brevity

Descriptive names aid understanding more than short ones

camelCase vs snake_case

No clear winner — consistency matters more than style



Style	Comprehension Score
x, t, i	❌ Low
tmp, val, initPt	⚠️ Medium
totalSales, getInvoiceAmount()	✅ High

“🧪 Lawrie et al.: Full-word identifiers led to 25–35% faster comprehension”

4-Factor Model for Identifier Readability (R(N))

Objective

Quantify how readable an identifier is

- **SC**: Semantic Clarity
- **ST**: Stylistic Adherence
- **LN**: Length Appropriateness
- **NL**: Natural Language Readability

Formula

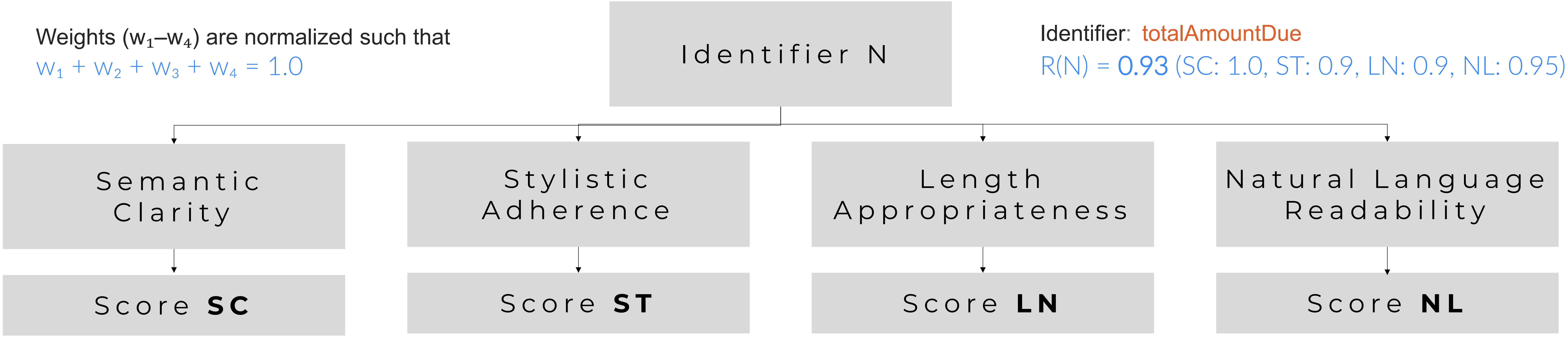
$R(N) = w_1 \cdot SC + w_2 \cdot ST + w_3 \cdot LN + w_4 \cdot NL$

Weights Calibrated

Based on expert ratings + regression over survey + code dataset

Weights (w_1–w_4) are normalized such that
$w_1 + w_2 + w_3 + w_4 = 1.0$

Identifier: totalAmountDue
$R(N) = 0.93$ (SC: 1.0, ST: 0.9, LN: 0.9, NL: 0.95)



Identifier N

Semantic Clarity

Stylistic Adherence

Length Appropriateness

Natural Language Readability

Score **SC**

Score **ST**

Score **LN**

Score **NL**

Weighted Sum → **$R(N) = w_1 \cdot SC + w_2 \cdot ST + w_3 \cdot LN + w_4 \cdot NL$**

Validating the Readability Model: Evidence from Real-World Code

Dataset

- 15 open-source projects
- **Languages**: Java, Python, JavaScript
- **1 million** lines of code analyzed

Validation Results

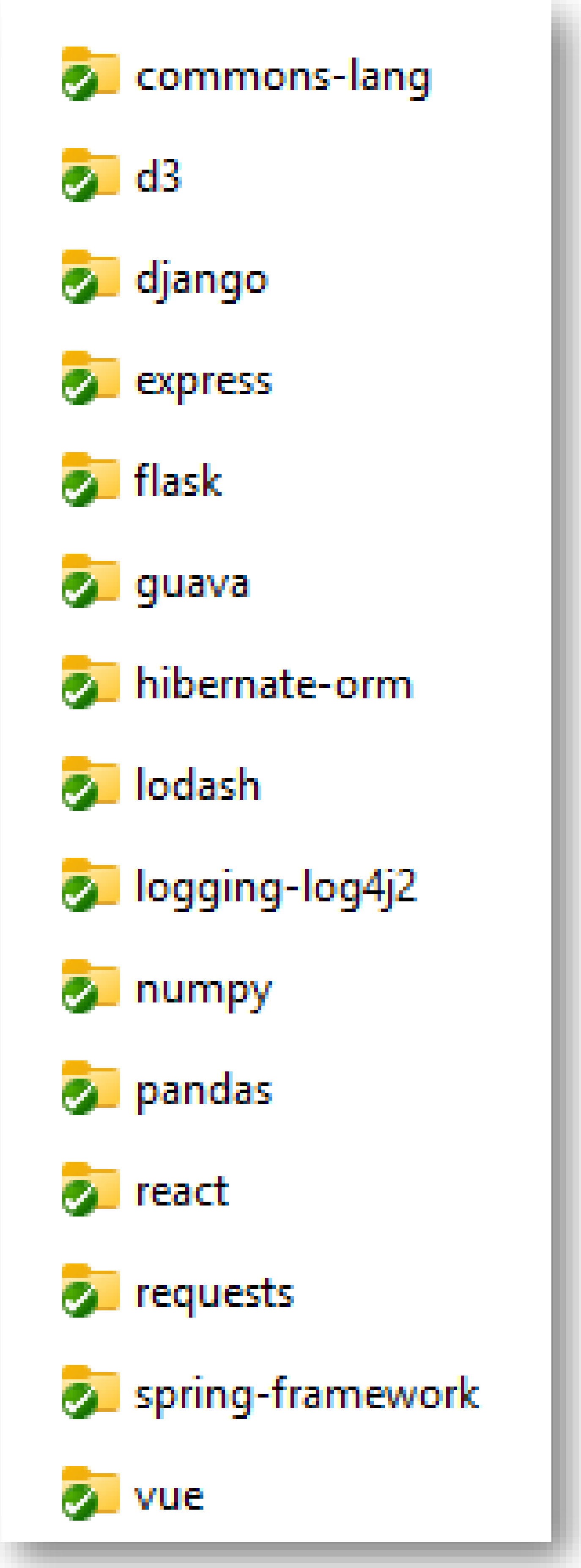
- High-scoring identifiers **aligned with expert intuition**
- Low-scoring identifiers **correlated with**:
  - Bug-prone modules
  - Poor onboarding experience
  - Negative code review comments

Case Studies

- Cleanup of low-score names → measurable improvements in:
  - Comprehension scores
  - Onboarding time
  - Maintenance activity

Identifier Readability Scores – Before and After Cleanup

Identifier	SC	ST	LN	NL	R(N)
x1rdy	0.1	0.2	0.5	0.1	0.23 ✗
retryCount	1.0	1.0	0.9	0.95	0.97 ✓



commons-lang
d3
django
express
flask
guava
hibernate-orm
lodash
logging-log4j2
numpy
pandas
react
requests
spring-framework
vue

Applying the Readability Model: From Code to Tools

For Developers

Integrate readability scores in code review, PR feedback, and linting tools.

For Researchers

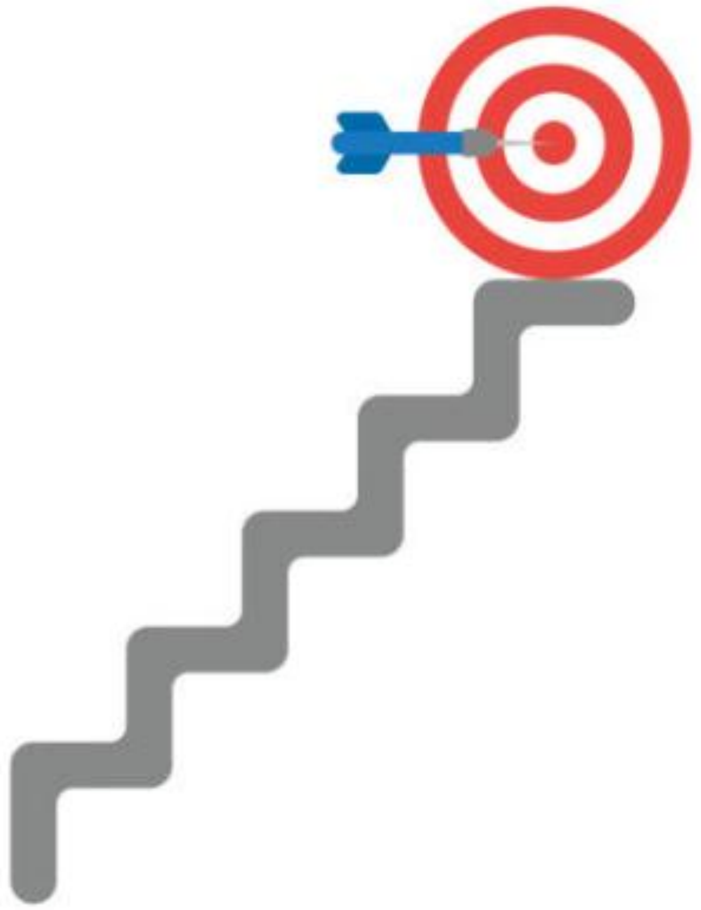
Enables studies on naming, cognition, and LLM-based name generation.

For Toolmakers

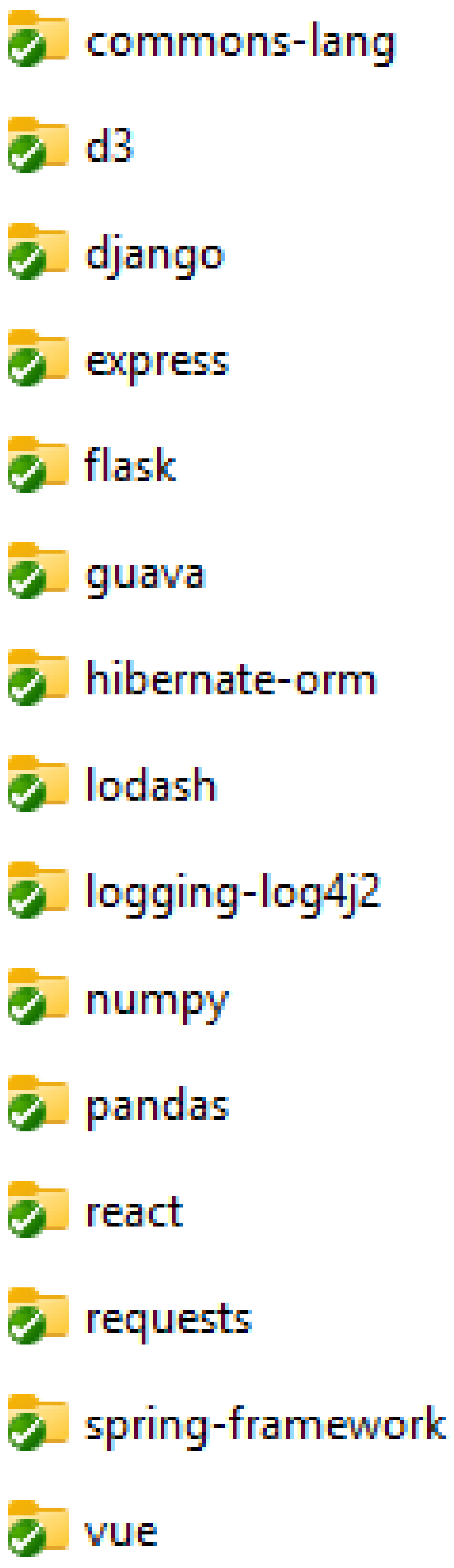
Embed model in IDEs or CI workflows to guide naming improvements.

Next Steps

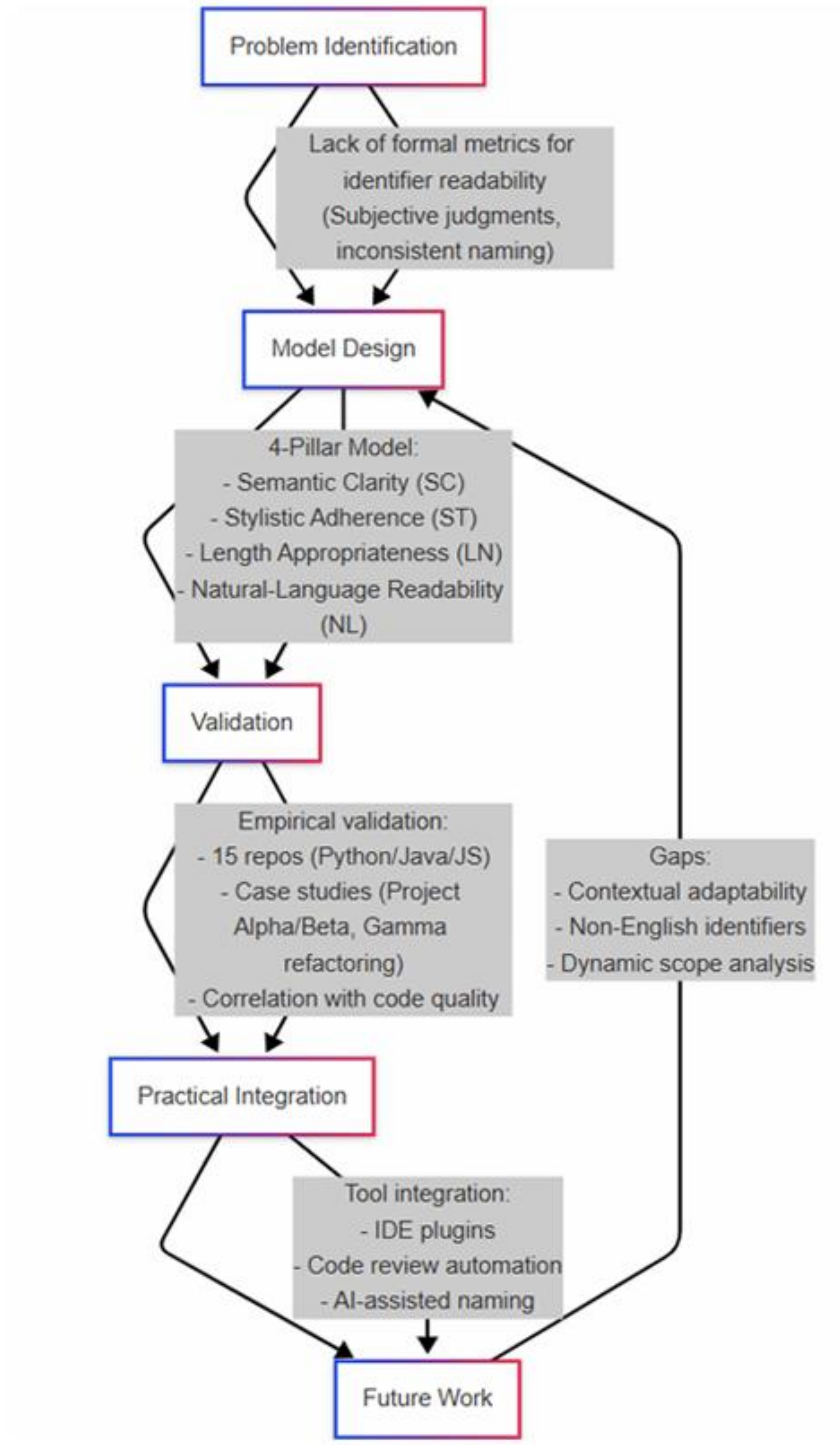
- Add context-aware scoring (e.g., based on usage scope and type)
- Integrate with refactoring suggestions in IDEs
- Evaluate LLM-generated identifiers using the model
- Use scores to train ML models for automatic renaming



ANNEXTURE



commons-lang
d3
django
express
flask
guava
hibernate-orm
lodash
logging-log4j2
numpy
pandas
react
requests
spring-framework
vue



Problem Identification

Lack of formal metrics for identifier readability (Subjective judgments, inconsistent naming)

Model Design

4-Pillar Model:
- Semantic Clarity (SC)
- Stylistic Adherence (ST)
- Length Appropriateness (LN)
- Natural-Language Readability (NL)

Validation

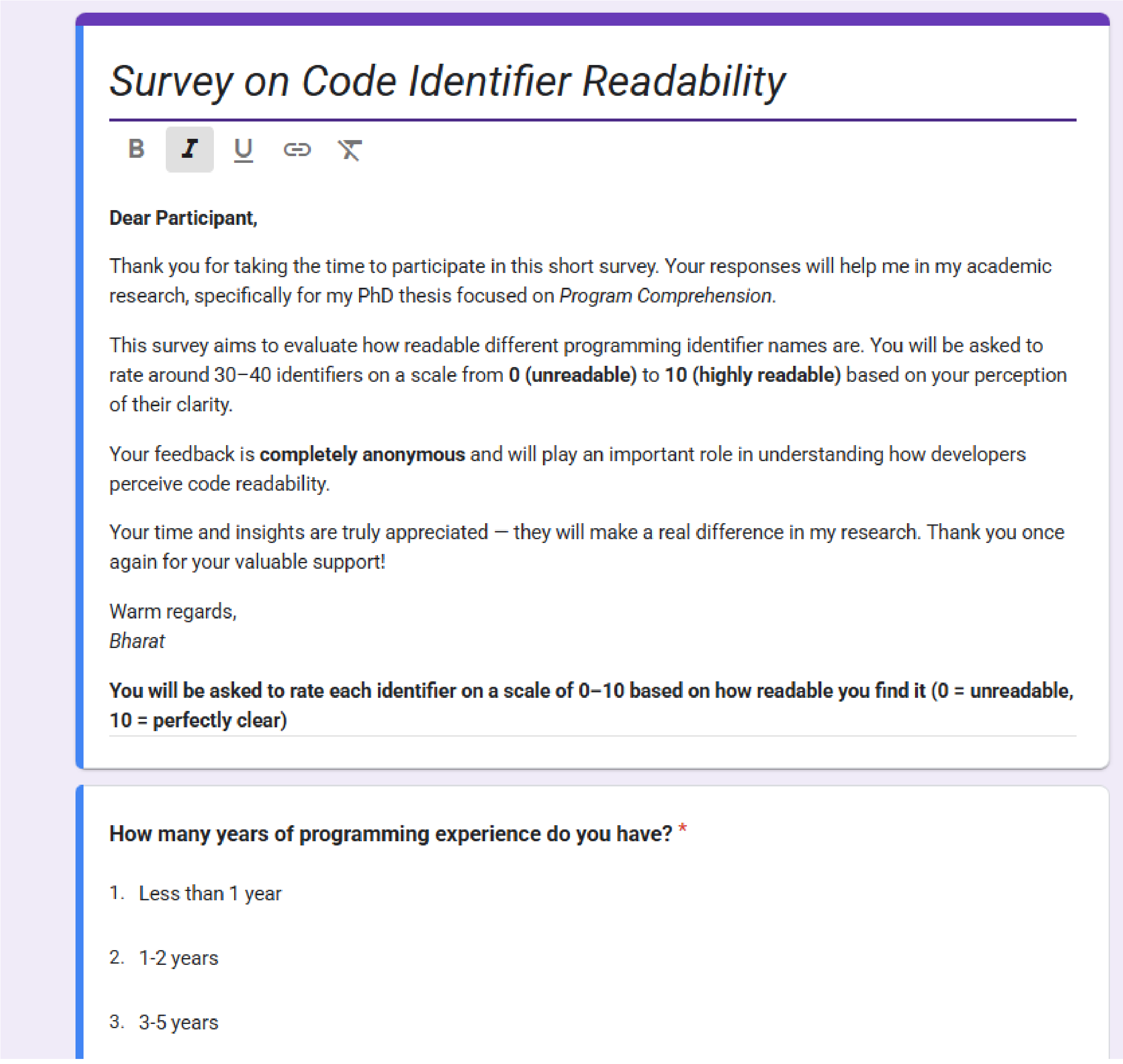
Empirical validation:
- 15 repos (Python/Java/JS)
- Case studies (Project Alpha/Beta, Gamma refactoring)
- Correlation with code quality

Practical Integration

Tool integration:
- IDE plugins
- Code review automation
- AI-assisted naming

Future Work

Gaps:
- Contextual adaptability
- Non-English identifiers
- Dynamic scope analysis



Survey on Code Identifier Readability

Dear Participant,

Thank you for taking the time to participate in this short survey. Your responses will help me in my academic research, specifically for my PhD thesis focused on *Program Comprehension*.

This survey aims to evaluate how readable different programming identifier names are. You will be asked to rate around 30–40 identifiers on a scale from **0 (unreadable)** to **10 (highly readable)** based on your perception of their clarity.

Your feedback is **completely anonymous** and will play an important role in understanding how developers perceive code readability.

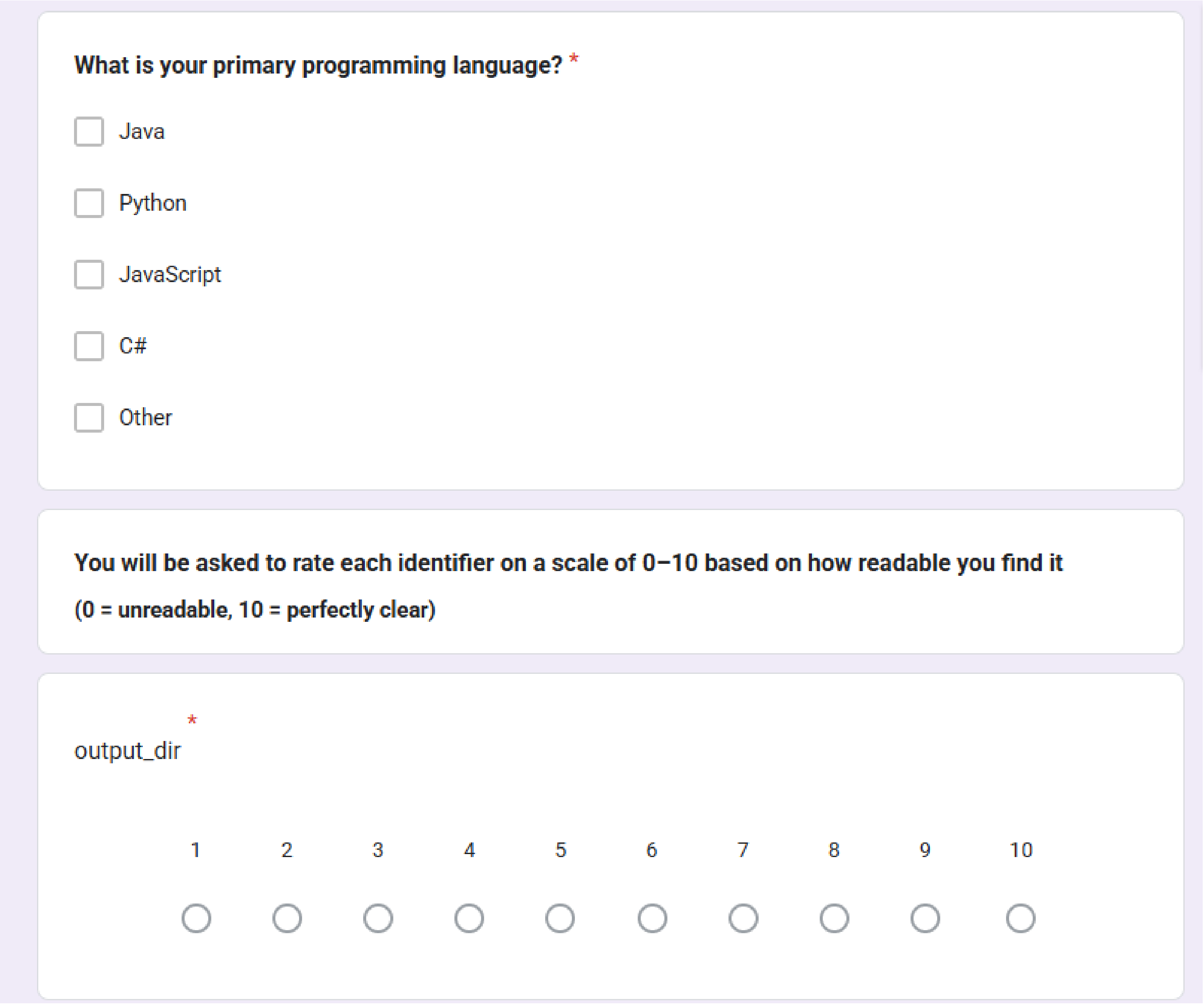
Your time and insights are truly appreciated — they will make a real difference in my research. Thank you once again for your valuable support!

Warm regards,
Bharat

You will be asked to rate each identifier on a scale of 0–10 based on how readable you find it (0 = unreadable, 10 = perfectly clear)

How many years of programming experience do you have? *

1. Less than 1 year
2. 1-2 years
3. 3-5 years



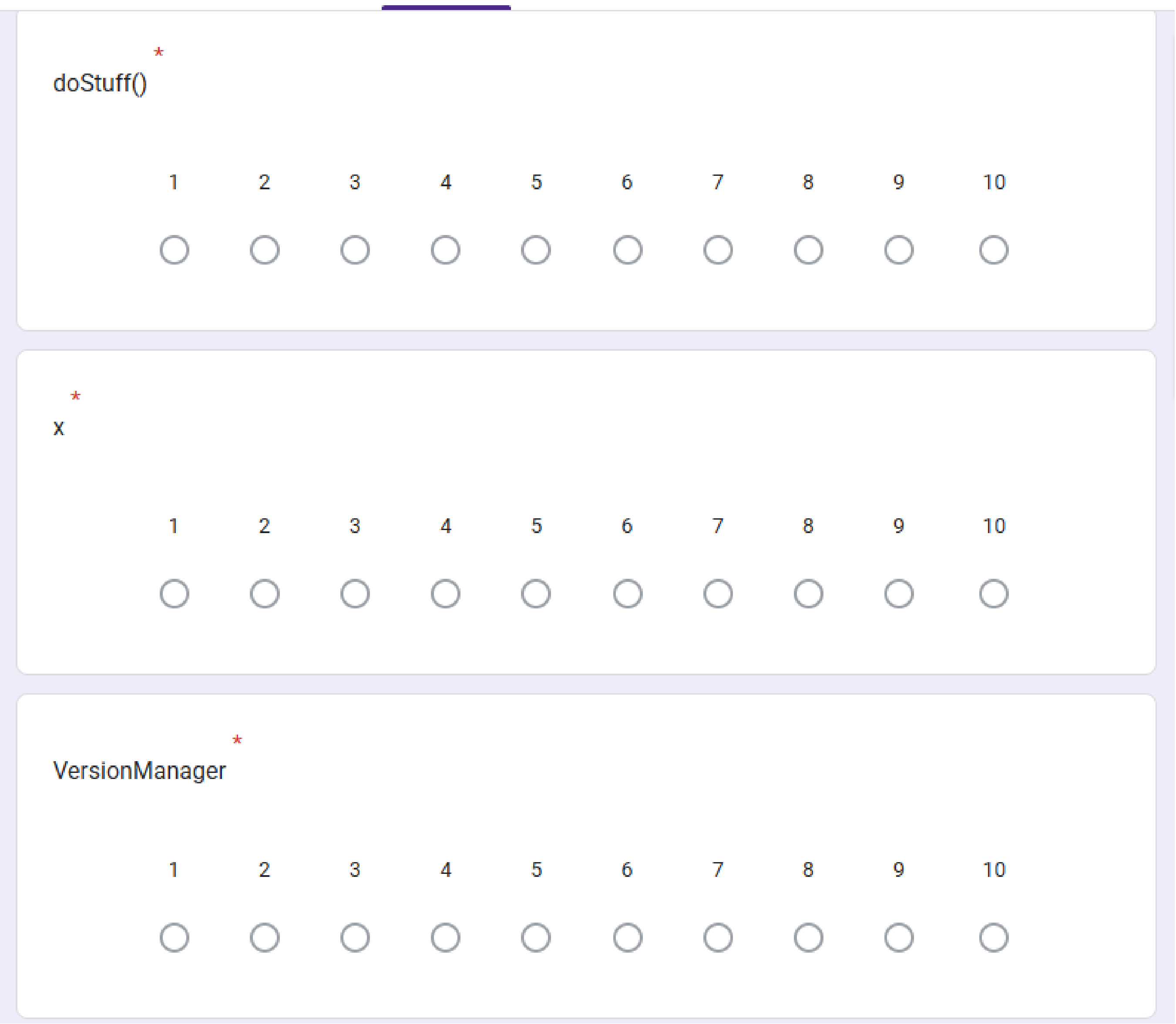
What is your primary programming language? *

- ☐ Java
- ☐ Python
- ☐ JavaScript
- ☐ C#
- ☐ Other

You will be asked to rate each identifier on a scale of 0–10 based on how readable you find it
(0 = unreadable, 10 = perfectly clear)

output_dir *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○



doStuff() *

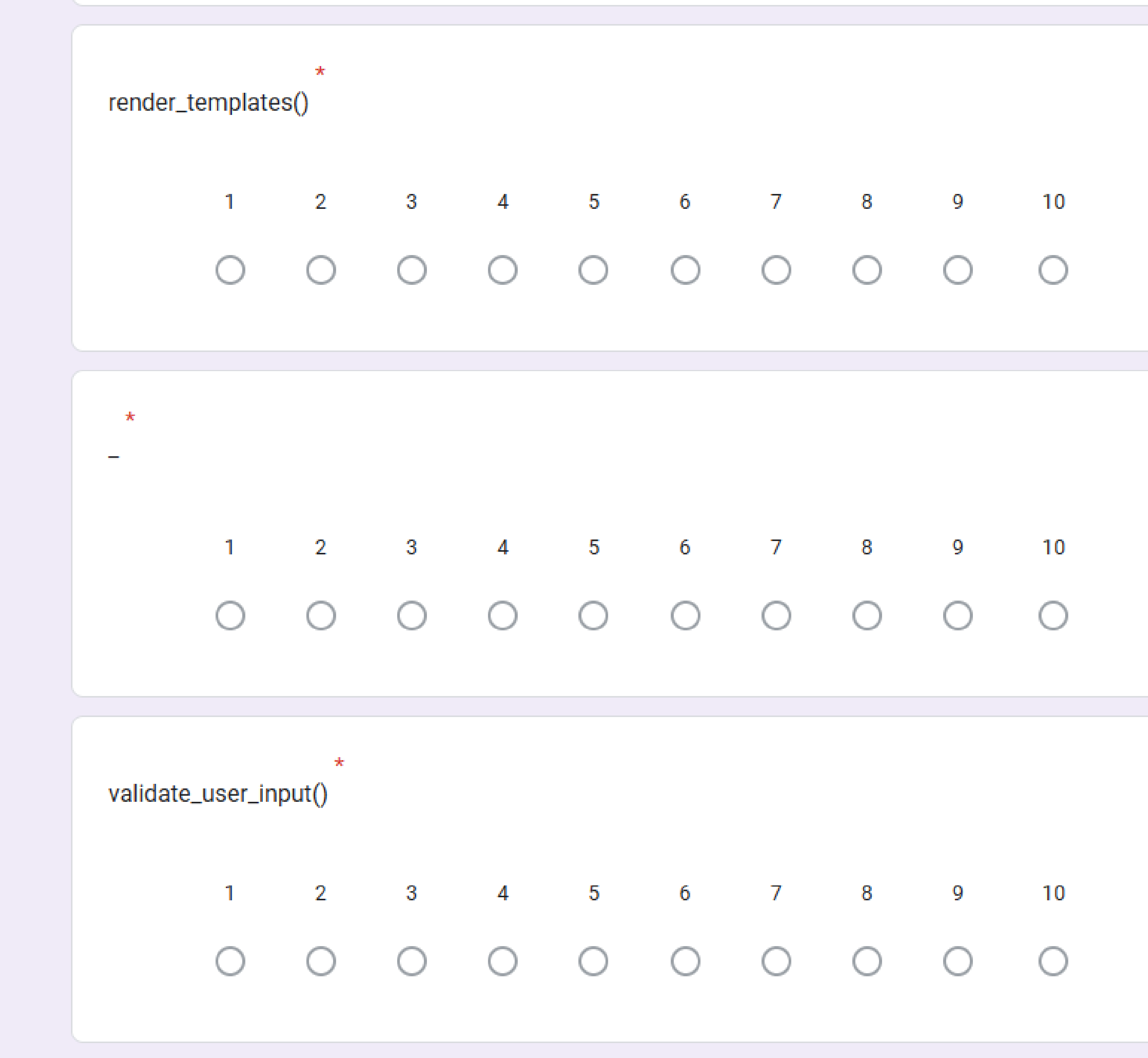
1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

x *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

VersionManager *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○



render_templates() *

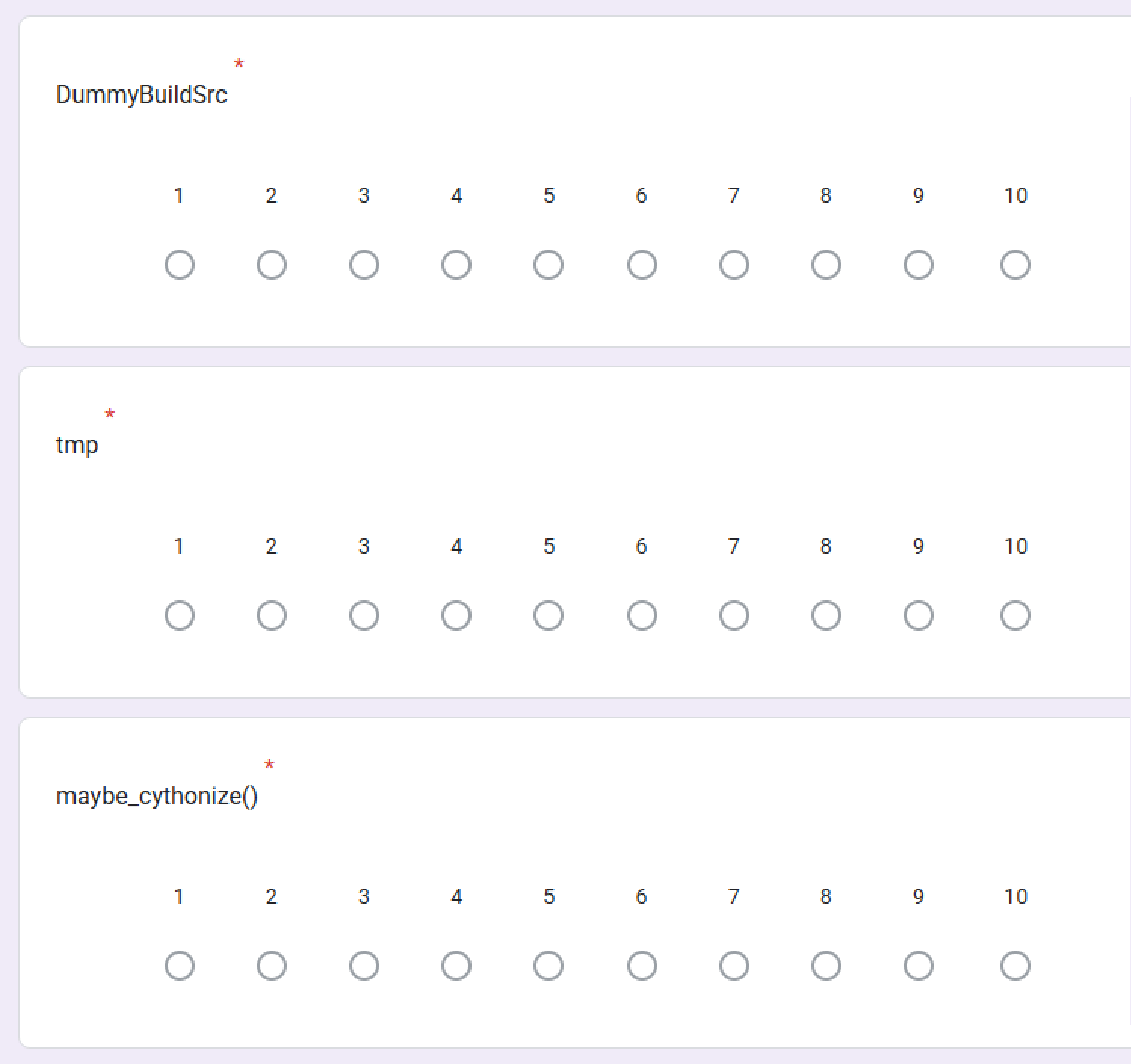
1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

_ *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

validate_user_input() *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○



DummyBuildSrc *

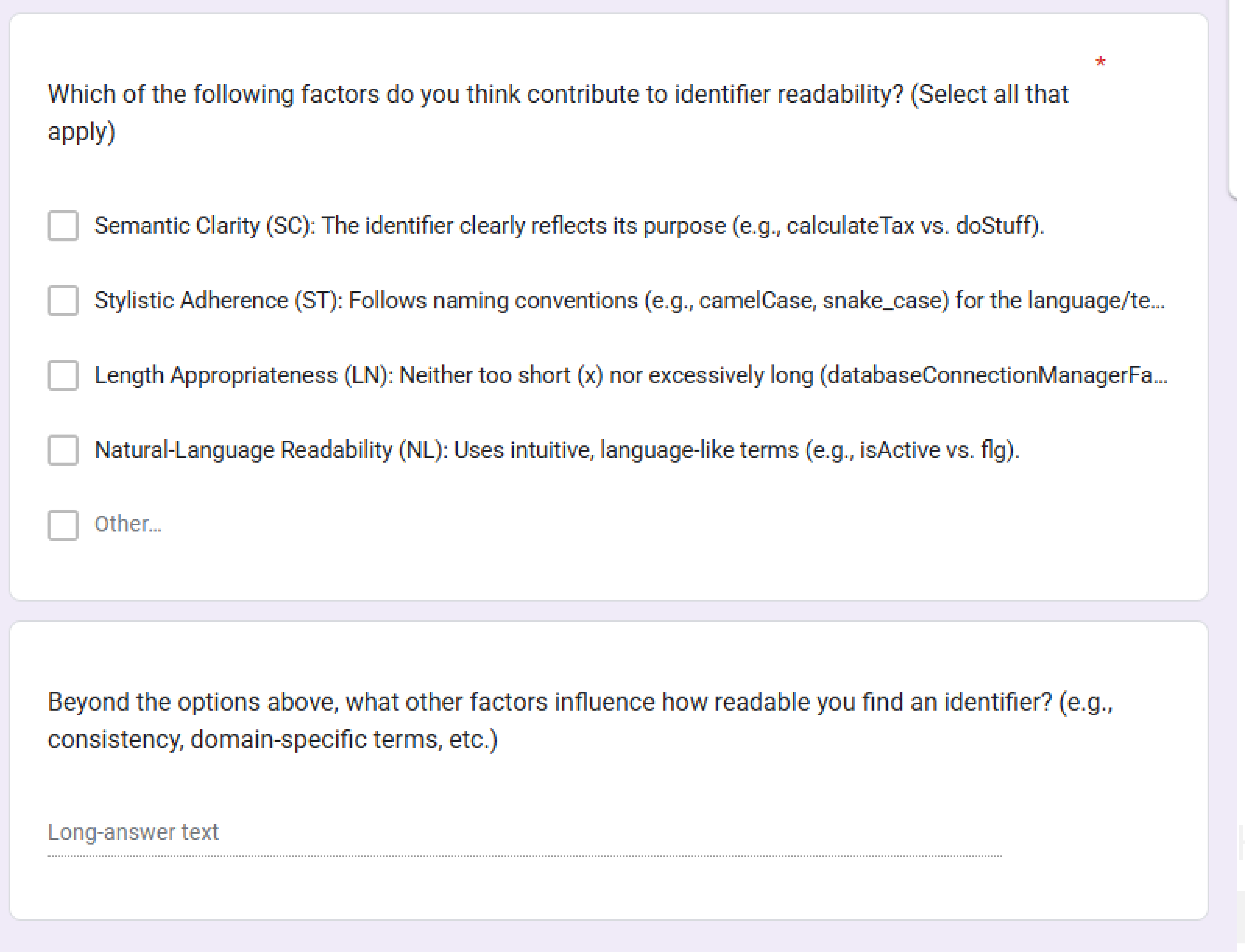
1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

tmp *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○

maybe_cythonize() *

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○



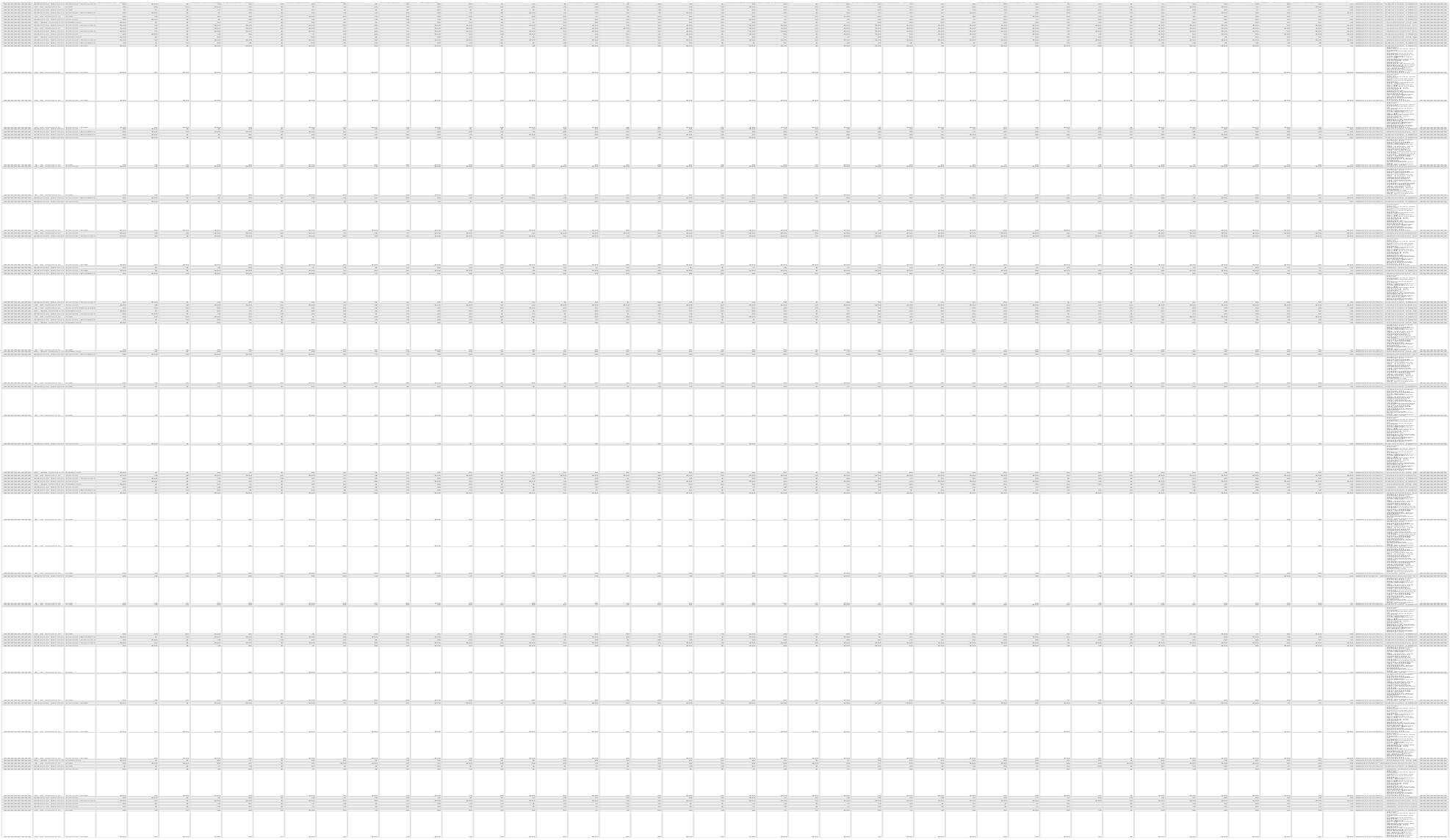
Which of the following factors do you think contribute to identifier readability? (Select all that apply) *

- ☐ Semantic Clarity (SC): The identifier clearly reflects its purpose (e.g., calculateTax vs. doStuff).
- ☐ Stylistic Adherence (ST): Follows naming conventions (e.g., camelCase, snake_case) for the language/te...
- ☐ Length Appropriateness (LN): Neither too short (x) nor excessively long (databaseConnectionManagerFa...
- ☐ Natural-Language Readability (NL): Uses intuitive, language-like terms (e.g., isActive vs. flg).
- ☐ Other...

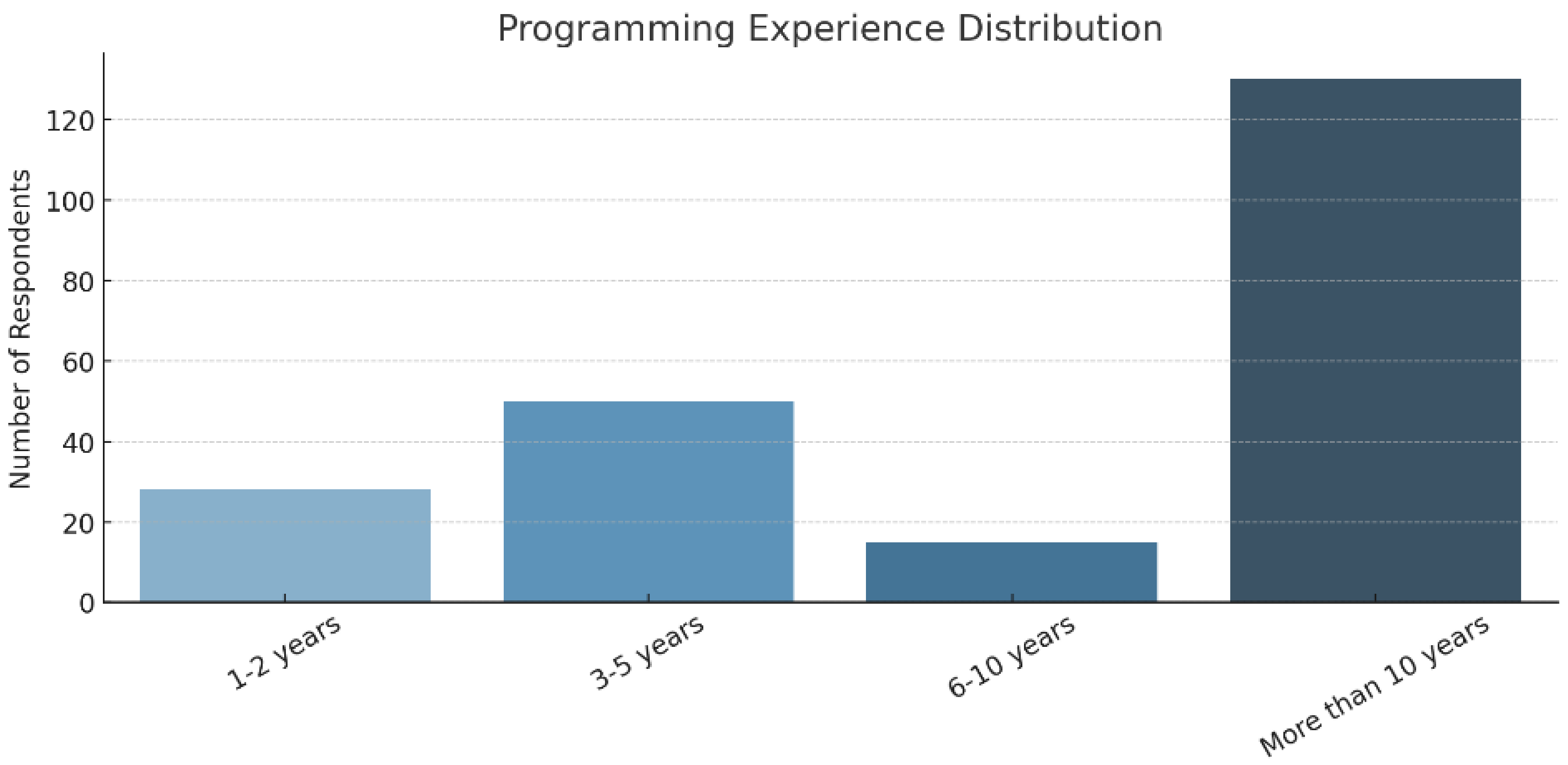
Beyond the options above, what other factors influence how readable you find an identifier? (e.g., consistency, domain-specific terms, etc.)

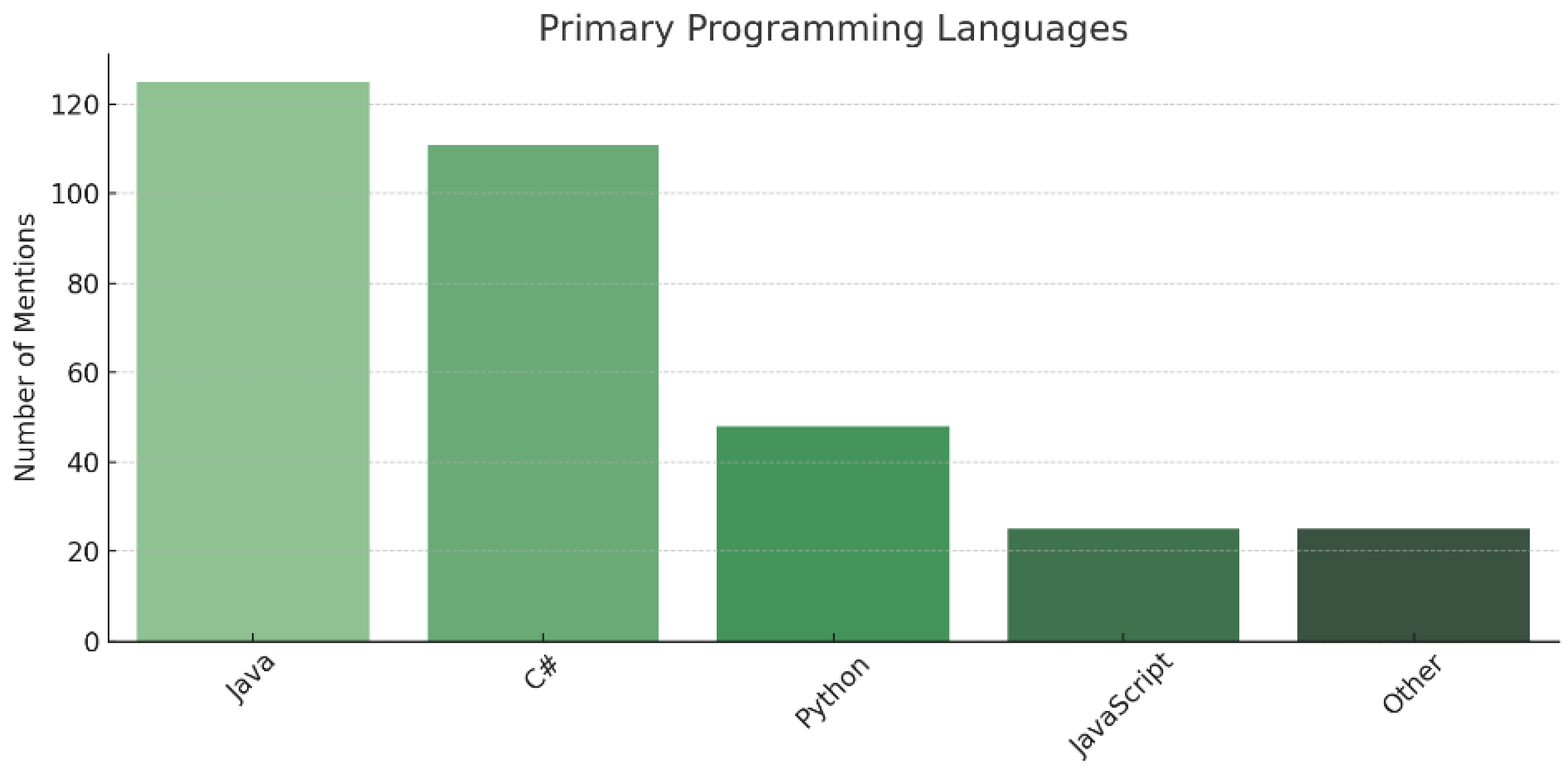
Long-answer text

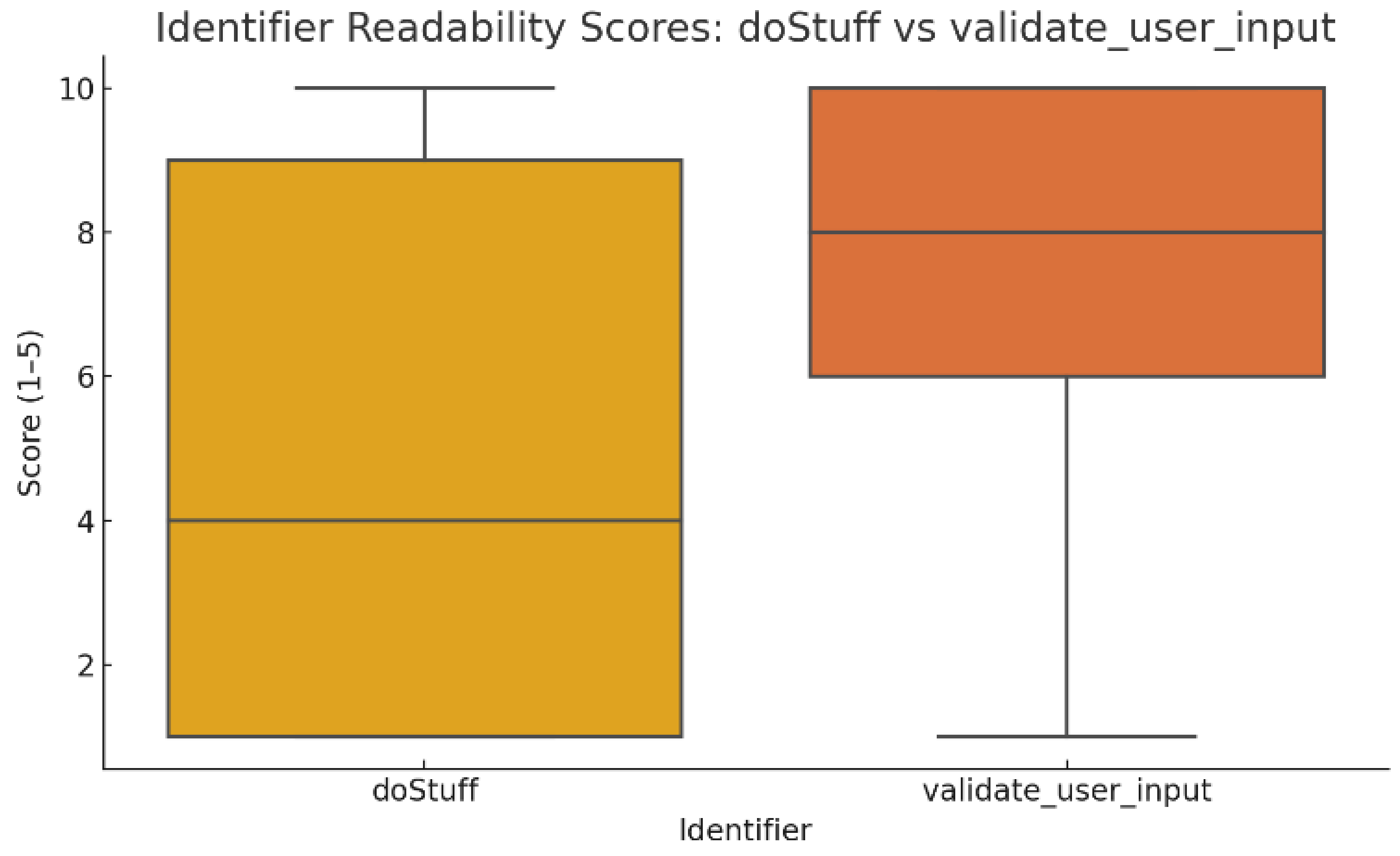
Identifier	Identifier	Identifier	Identifier
output_dir	is_debug_mode	InputValidator	linetrace()
doStuff	Factorize	testIsInitialized()	parsed
x	i, j, k	d	f()
VersionManager	get()	calculate_tax()	user_data
render_templates()	Hashing	LazyInitializerTestImpl	CleanCommand
_	write_version_info()	clean_me()	initialize()
validate_user_input	process_tempita()	macros	version_info
DummyBuildSrc	ext_data	parse()	db_query()
tmp	run()	MemoizerComputableTest	ConfigParser
maybe_cythonize()	pxifile	setup_cache()	handle()

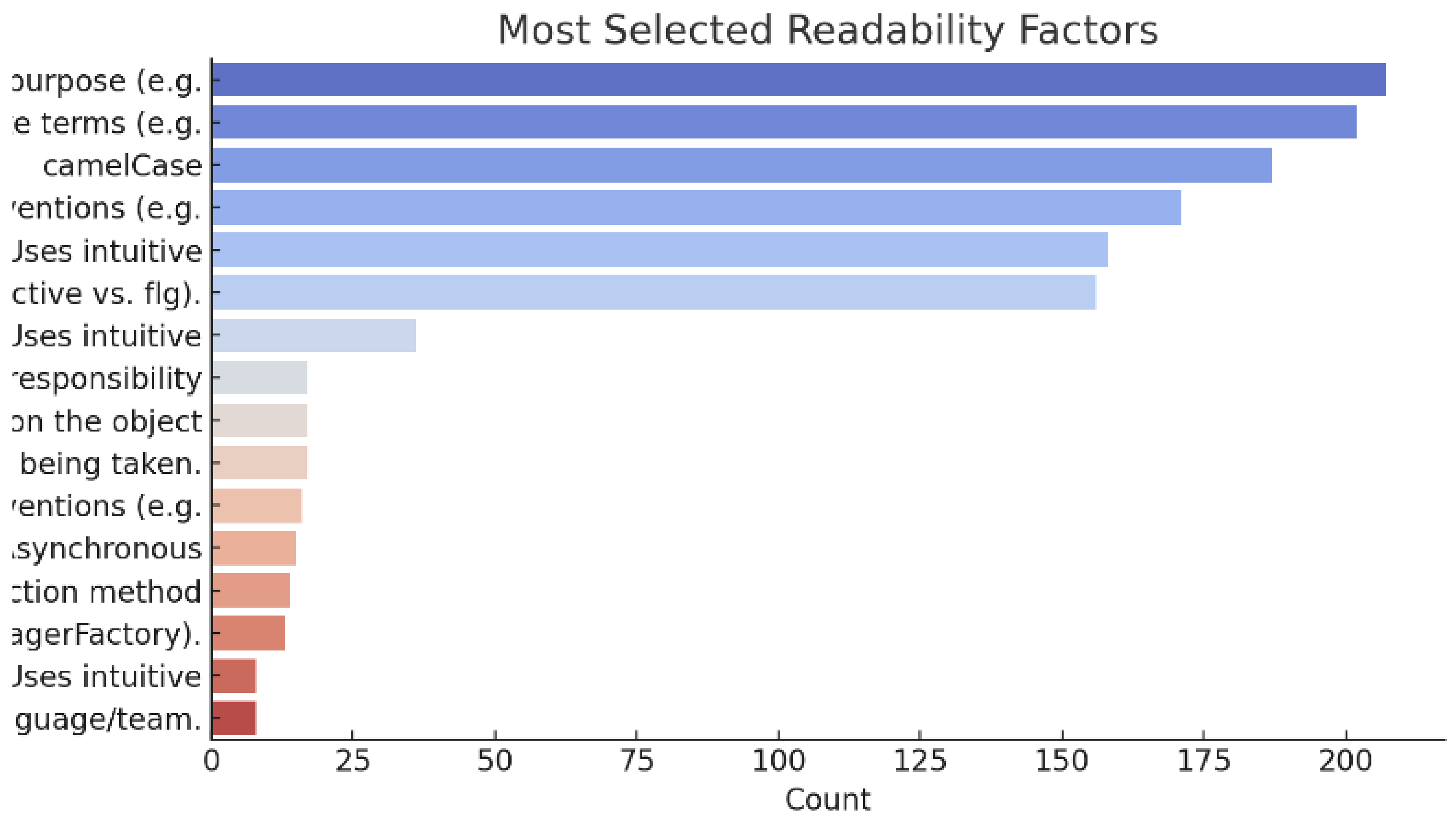



code works
identifiers consistently
line screen
descriptive identifiers
ght
important
trade
naming
conventi
x algorithm
class etc
sacrifice c
hm Domain
variable function
screen width
Boolean
Name
poor
fine
depend
often
naming style
g id
two Use
isActive isEnabled
g a
abbreviation
strike
widely accepted
button exces
loop counter
g isActive
clarity Avoiding
Domain
Specifi
hinder readability
variable representing
readable Consistency
vital Context
rent type
need
Consistency
negative ones
avoid confusion
easy
identifier names
time identifier
project
definitely makes
always matte
style e
positive boolean
documentation
well understood
significantly reduce
identifier btn
shorter identifiers
Acronyms Similar
naming th
use
purpose
purpose Consistency
softwar
confusion proper
logic easier
prefix
quicker
g
abbreviations acronyms
specific names
always using
developers working
mmon acronyms
ensure
take
avoided
functional naming
Usi
perfectly acceptable
unambiguous Pronounceability
one p
readability
spelled
specific
ones e
fetchUserInfo elsewhere
context
reduce readability
methods
brevity
language alwa
discuss
width Consistency
avoid unnecessary
clearly indicate
suffix
example
greatly enhance
readability Clarity
follow Clarity
vs Brevity
makes
cont
Domain langua
use fetchUserInfo
variable
use positive
classes functional
refer
unfamiliar
helps developers
names e
Pronounceability Identifiers
abbreviations sparingly
indirectly contribute
inde
identifier
common ones
proper use
getUserData
e
bala
accepted e
Terms Using
remember
generally
less common
understood
Intent
function class
used
code
pronounce
codeb
naming avoid
type avoid
names avoid
consistently throughout
using ca
oiding Negation
readable
type
crucial
caution Common
Consistency throughout
Abbreviations
developers understand
obscure abbreviations
isEnabled rather
generally better
isNotActive isDisabled
important Consistency
co


Programming Experience Distribution


Primary Programming Languages


Identifier Readability Scores: doStuff vs validate_user_input


Most Selected Readability Factors