

Project Name	AI Driven Testing
Online team meeting	https://fau.zoom.us/j/69553438847?pwd=VHGm0oOZ2KvPbbhrGsmX83d4rAuuaT.1
Production system (if any)	...
Test system (if any)	...
GitHub repository	https://github.com/amosproj/amos2025ss04-ai-driven-testing
GitHub feature board	https://github.com/orgs/amosproj/projects/82/views/2
GitHub imp-squared backlog	https://github.com/orgs/amosproj/projects/86/views/1
Planning Poker Link	https://pokerplanning.org/room/05e2b4ba-b452-4c29-8ba0-e9ae23005ce0
Team T-shirt (white)	https://www.shirtinator.ch/s/gyMoSd27QOSIIB0wUYQ7XA
Team T-shirt (black)	https://www.shirtinator.ch/s/-GuNOvW5Q2qjHFDZrYILpA
black link again since it didnt work for some	https://www.shirtinator.ch/t-shirts/gestalten/t-shirt-bedrucken#/load/share/f86b8d3a-f5b9-436a-a31c-50d9ad820ba4
Additional materials	...
Team mailing list	oss-amos-proj4@lists.fau.de

Last Name	First Name	GitHub User Name	Email Address
Brüggemann	Jonas	JonasBrue	jonas.brueggemann@campus.tu-berlin.de
Clicqué	Lennard	OlafVanHuusen	lennard.clicque@fau.de
Hasse	Lisabeth	PeppermintCoding123	lisabeth.hasse@fau.de
Heidkamp	Tessa	theidkamp	tessa.v.heidkamp@campus.tu-berlin.de
Krug	Maximilian	HaruspexSan	maximilian.krug@fau.de
Lang	Felix	xilef45	felix.l.lang@fau.de
Lorenz	Alexander	Hydraneut	alexander.lorenz@fau.de
Parameswaran	Birnavan	Birnavan-Parameswaran	parameswaran@campus.tu-berlin.de
Takale	Aditi Vishwas	adititakale01	adititakale01@gmail.com
Alsultan	Moaiad	Moaiadsu	m.alhmadhalsultan@campus.tu-berlin.de

#	Meeting Day	Product Owner		Software Developer	Release Manager	Scrum Master	First-Level-Support :)	Comment
		Review	Planning					
1	2025-04-16	both	both	all other	none	Felix	Felix	
2	2025-04-23	both	Max	all other	Max	Felix	Felix	
3	2025-04-30	Max	Alex	all other	Alex	Felix	Felix	
4	2025-05-07	Alex	Max	all other	Jonas	Felix	Felix	Build process review
5	2025-05-14	Max	Alex	all other	Lennard	Felix	Felix	
6	2025-05-21	Alex	Max	all other	Lisabeth	Felix	Felix	
7	2025-05-28	Max	Alex	all other	Tessa	Felix	Felix	Mid-term due
8	2025-06-04	Alex	Max	all other	Max	Felix	Felix	
9	2025-06-11	Max	Alex	all other	Alexander	Felix	Felix	
10	2025-06-18	Alex	Max	all other	Biranavan	Felix	Felix	
11	2025-06-25	Max	Alex	all other	Aditi Vishwas	Felix	Felix	
12	2025-07-02	Alex	Max	all other	Moaiad	Felix	Felix	
13	2025-07-09	Max	Alex	all other		Felix	Felix	
14	2025-07-16			all other		Felix	Felix	Demo day!
Product owners, software developers, and Scrum Master are set and ideally don't change over time; the critical part is the Release Manager role you need to define here								

Goals	Interpersonal relationship objectives: "To foster an atmosphere of mutual respect and learning, creating a team-spirit"
	Main goal is to create a satisfying project that makes Us, the IP(Industry Partner) and open-source community happy
Meeting norms	Documentation in english and meeting language in German
	Start on time
	Be Polite
	Purpose: Clearly define the goal of the meeting. Preparation: Ensure all participants are well-informed and ready to contribute. Participation: Encourage active engagement from all attendees. Process: Organize the meeting structure and agenda effectively. Progress: Monitor the meeting's progress and ensure it stays on track.
Working norms	Leave the code better than you found it (The Boy Scout Rule)
	Keep it Simple, Stupid (KISS)
Coordination norms	Felix will do the retros + keeping track of meetings
	Alex and Max will take care of the assignments (Backlog Items)
Communication norms	We check Discord at least every day
Consideration norms	Communicate Conflict in Team-Meeting. Help from Felix or Jovana to solve conflicts.
	Respect, active listening, tolerance
Cont. improvement norms	Experimentation spirit and feedback culture
	Burn-Down charts as process tracking
Rewards	Appreciation and praise
Sanctions	Wear the ducky tie
	Post a cute/funny pic of a pet (or similar)
Signatures	
Scrum Master	Felix Lang
Product owner	Maximilian Krug
Product owner	Alexander Lorenz
Software developer	Mohammad Moaiad Alhamdh Alsultan (Moaiad Alsultan)
Software developer	Jonas Brüggemann
Software developer	Lennard Clicqué
Software developer	Birnavan Parameswaran (Biri)
Software developer	Lisabeth Hasse
Software developer	Tessa Heidkamp
Software developer	Aditi Vishwas Takale

Product Vision	Project Mission
<p>Product Vision</p> <p>We are transforming the landscape of software testing through intelligent, AI-powered automation.</p> <p>Our vision is a future where testing is no longer a bottleneck but a seamless, continuous, and reliable part of the development process. By empowering developers and testers with AI tools that autonomously generate and maintain test cases, we free them from generating the same test cases over and over to instead focus on creative problem-solving and strategic innovation. Our solution supports a future of scalable, secure, and locally controlled quality assurance, driving the next evolution in software development. We envision the role of the programmer to become a curator and innovator of AI generated Code.</p>	<p>Product Mission</p> <p>Deliver a minimum viable product (MVP) of an AI-powered test generation assistant that interacts effectively with real-world software projects. This includes laying the empirical and research groundwork for the idea. From this research we will conclude a MVP, that can analyze simple Python and C++ programs and produce corresponding test cases. Operated through a user-friendly chat interface. The MVP will showcase initial integration with widely used open-source testing frameworks such as Robot Framework, proving the viability of AI-assisted test automation in practical, everyday development environments.</p>

Term	Definition
LLM	Large language model
LLM Input	User Input into the LLM
LLM Output	Response of the LLM to the Input
LLM System Prompt	System message sent (this is different from the LLM Input)
Temperature	gradient how "creative" the LLM behaves
Interface Script/ LLM factory	Backbone script of the project to start, end and use the LLM containers
Container	Docker container, (not kaniko os something else)
IP	Industry Partner
Sprint	agile (Amos) sprint with a duration of 2 weeks
CI	GitHub workflows as Continuouse Integration assistant
Wiki	Documentation in the Github repos wiki (not Wikipedia)
Laptop	Personal device to run the code (nothing fancy standard device) TODO add specs

Sprint #	Sprint goal
1	None
2	None
3	None
4	Laying foundational architecture groundwork
5	Initialize future development area
6	Enhance user value by metricization
7	Preparing for Midproject review
8	Expanding Modularity and advancing metricization / Refactoring
9	Enhancing accuracy of LLM return
10	Advancing Usability and LLM response statistics
11	Design and Development for own LLM
12	Create CI Usecase and user value
13	Prepare for demo day
14	Last touches before the finally
15	-/- (only winter term)

Sprint #	Story Points Planned	Story Points Realized
1		
2	33	36
3	22	20
4	23	23
5	27	28
6	21	25
7	54	24
8	31	
9		
10		
11		
12		
13		
14		
15		
	PLEASE CREATE THE VELOCITY CHART ON A NEW TAB USING THE DATA FROM THIS TAB	

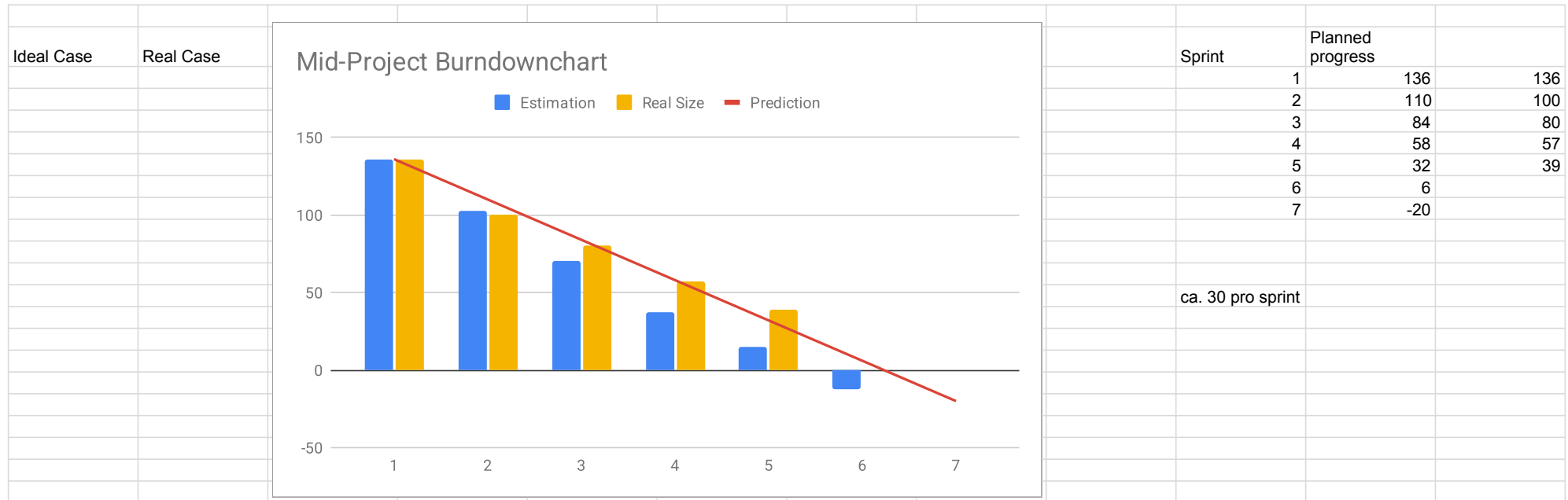


Sprint	Goal	Feature Name	Est. size	Est. remaining	Real size	Real remaining
Release						
Total			136	136	132	
Sprints						
1			0	136	0	132
2			33	103	36	132
3			33	70	20	96
4			22	37	23	76
5			27	15	28	53
6			21	-12	25	25
7						0
8						0
9						0
10						0
11						0
12						0
13						0
14						0
15						0
Finish						0
Features						
1						
		(none since we didnt meet with the IP yet)				
2						
		Write .py script to interact with the LLM 3	3		3	
		Reaserch LLM 5	5		5	
		Research Deepcoder/Phi4-Mini	5		5	
		Reaserch LLM 4	5		5	
		Reaserch LLM 2	5		8	

Sprint	Goal	Feature Name	Est. size	Est. remaining	Real size	Real remaining
		Create initial Architecture Documentation	2		2	
		Research LLM 1	5		3	
		Write .py script to interact with the LLM 1	3		5	
3						
		Continue to Maintain the Architecture-Documents	3		3	
		Write .py script to interact with the LLM 2	3		3	
		Research Code Complexity	5		5	
		Create Python test cases	3		1	
		Start Bill of Materials	1		1	
		Continue to Maintain Bill of Materials	1		1	
		Write .py script to interact with the LLM 5	3		3	
		Write .py script to interact with the LLM 4	3		3	
4						
		Continue to Maintain Bill of Materials	1		1	
		Unify the python interface	3		3	
		initialize ReadMe	2		1	
		Research AI-Model Benchmark	3		3	
		Build the first Benchmarking tests	3		3	
		make CI-Pipeline	5		8	
		Continue to Maintain the Architecture-Documents	3		2	
		Set up Branchmanagement	3		2	
5						
		Add Modul Interface	5		5	
		Research Ollama	5		3	
		Test docker performance	3		5	
		Initialize API	5		5	
		Build process video	3		3	
		Initialize Frontend	5		3	
		Continue to Maintain Bill of Materials	1		1	
		Make metrics.py into its own modul			3	
6						
		Write .py script to start docker containers	3		3	
		Show POs how the Project works	3		2	
		Research: Spell checker	3		3	
		Onboard small LLM	1		1	
		Frontend Make User enter prompt and source code	5		8	
		Module: Analyze context size	3		5	
		Connect API to Frontend	3		3	

[illegible]

[illegible]



Sprint	Goal	Feature Name	Est. size	Est. remaining	Real size	Real remaining
Release						
Total		estimation ca. 30 per sprint	226	226		
Sprints						
7			54	226	24	226
8			32	172	0	202
9			31	140	0	202
10			25	109	0	202
11			22	84	0	202
12			24	62	0	202
13			23	38	0	202
14			15	15	0	202
Features						
7						
		Module. Evaluate LLM performance	3			
		Extract Code from Input/Output	3		2	
		Research/Implement How to include whole project	5		13	
		Module: Code Complexity MCC	5			
		Set up Docker compose	5		5	
		Unify build process	5			
		Bill of Material	1		1	
		Architecture Document	3		3	
		Module CCC	5			
		Module: external research	5			
		Initialize user, (technical) design, and build/deploy documentation	3			
		Refactoring: unify payload to json	5			
		Module: CMT Code Complexity	3			
		Set prerequisites for ModuleI	3			
8						
		Module. Evaluate LLM performance	3			
		Module: Code Complexity MCC	5			
		Module CCC	5			
		Bug-Fix: Update Extract Code from Input/Output #128	1			
		Refactoring: unify payload to json	5			
		Unify build process	5			
		Set prerequisites for ModuleI	3			

Sprint	Goal	Feature Name	Est. size	Est. remaining	Real size	Real remaining
9		Onboard and Test new LLM	2			
		Initialize user, (technical) design, and build/deploy documentation	3			
		Onboard and Test new LLM	3			
		Research: Befits of chaining methods	2			
		Module: LLM testing of Code understandability, make LLM describe the code, let the same LLM (with no history) build the code from the description -> see if code passes the Unit tests.	5			
		Research: How to measure code coverage with tools	5			
		Module: Flaky Test Identifier: Detect and flag non-deterministic/generated tests	5			
		Research: can we run the models in the CI	3			
10		Module: Show dependency hierarchy of code by logic flow	3			
		Module: Detect Edgecase accuracy of LLM research	5			
		Onboard and Test new LLM	3			
		Module: Allow chaining outputs between models	3			
		Module: Check output for code coverage of the tests	3			
		CI: create CI for running everything in CI	2			
		Frontend: make modules togglable in UI	5			
		Module: Add configurable timeouts for LLM generation	3			
11		Test current models for output consistency (deterministic)	3			
		Output exporter (as zip, json, http etc..)	3			
		Use metrics to compare 1b, 3b, 7b and maybe 14b models	2			
		Research: test readability score (Flesch/Kincaid)	3			
		Module: Detect redundant/duplicate test cases	3			
		Module: Auto-group similar test cases into suites	5			
		Module: Add security analyzer module for generated code	3			
		Module: Map language to optimal model with fallback support	3			
12		Research: how to train/ enhance own LLM on HPC	3			
		Module: Measure test readability score (Flesch/Kincaid)	5			
		Use HPC to train our own LLM	5			
		Try HPC to run the thing and test a larger model 50b+ parameters	5			
		Module: Auto-split large files into units for test generation	3			
		Research: different types of software test	3			
		Evaluate the performance of the models with and without the modules	3			
13						
		Homework: Prepare Demo and create 3 Usecases	2			
		Clean the wiki	2			

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

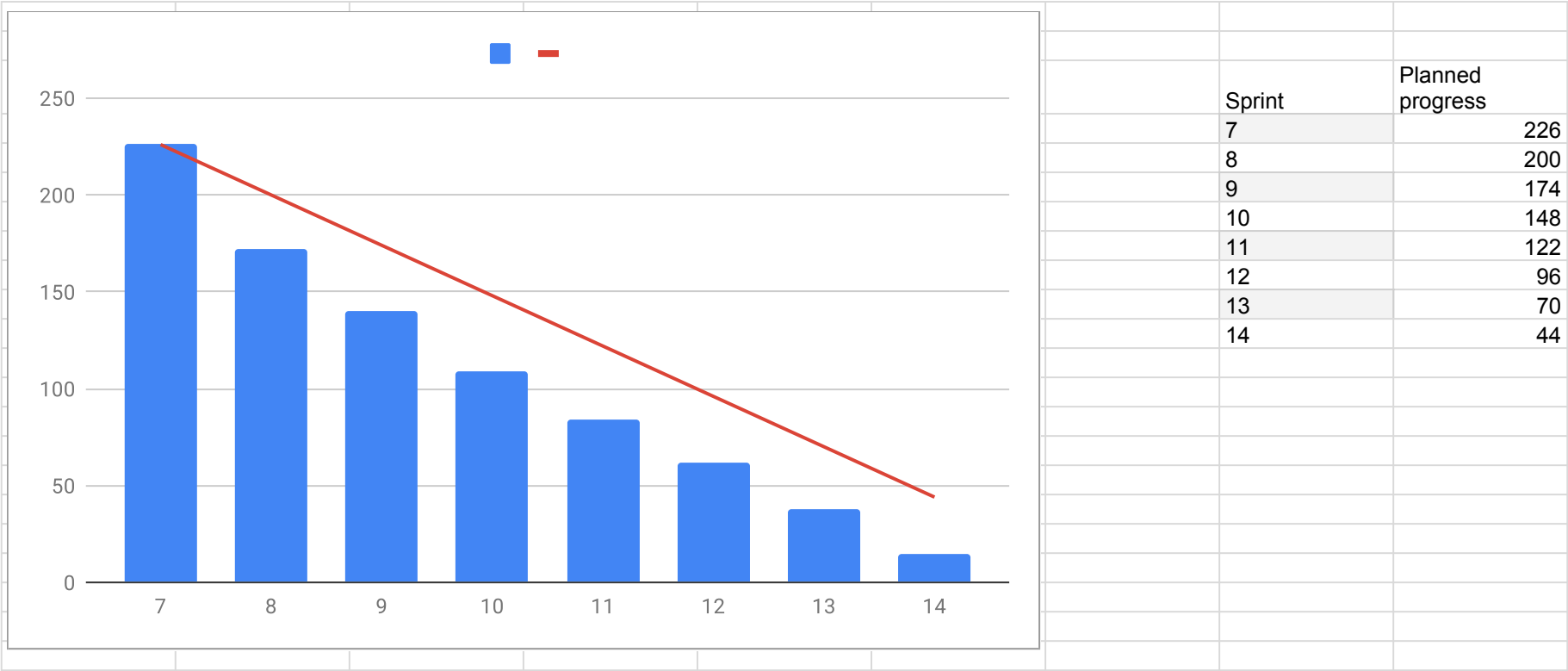
[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



[illegible]

Type	Link / reference

#	Context	Name	Version	License	Comment
	Python Version for running the main script	Python V	3.11	Python Software Foundation License Version 2	
	Running Ollama via Docker	Docker	28.1.1	Apache License, Version 2.0.	
	Ollama enables running LLMs easily	Ollama	0.6.8	MIT License	
	Running Qwen2.5-Coder via Ollama	Qwen2.5-coder	qwen2.5-coder:3b-instruct-q8_0	Qwen Research License (3B Model only), Apache License 2.0 (others)	
	Running Mistral via Ollama	Mistral	mistral:7b-instruct-v0.3-q3_K_M	Apache License, Version 2.0.	
	Running Deepseek-Coder via Ollama	Deepseek-Coder	deepseek-coder:6.7b-instruct-q3_K_M	Deepseek License Agreement	only used for testing
	Running Phi4-Mini via Ollama	Phi4-Mini	phi4-mini:3.8b-q4_K_M	MIT License	
	Running Gemma via Ollama	Gemma	gemma3:4b-it-q4_K_M	Gemma Terms of Use	only used for testing
	Running Tinyllama via Ollama	Tinyllama	tinyllama:1.1b	Apache License, Version 2.0.	
	Running Qwen3 via Ollama	Qwen3	qwen3:4b-q4_K_M	Apache License, Version 2.0.	
	mainly used language	python		PSF License	
	devDependency	pytest	8.3.1	MIT License	used for unit-tests
	Used python library	docker	7.1.0	Apache License, Version 2.0.	
	Used python library	requests	2.32.2	Apache License, Version 2.0.	
	Used python library	tqdm	4.67.1	MIT License	
	Used python library	numpy	1.22.0	BSD License	
	Used python library	pandas	1.3.0	BSD License	
	Used python library	fastapi	0.115.1	MIT License	
	Used python library	uvicorn[standard]	0.34.0	BSD License	
	Used python library	transformers	4.52.2	Apache License, Version 2.0.	
	Used python library	langchain	0.3.25	MIT License	
	environment management	conda	25.5.0	BSD-3 License	
	Distribution channel	conda-forge	25.5.0	BSD-3 License	
	frontend	react	19.1.0	MIT License	
	frontend	react-dom	19.1.0	MIT License	
	frontend	react-scripts	5.0.1	MIT License	
	frontend	typescript	4.9.5	Apache License, Version 2.0.	
	frontend	web-vitals	2.1.4	Apache License, Version 2.0.	
	Used for Code formatting	Black	25.1.0	MIT License	
	Used for Linting	flake8	7.0.0	MIT License	

Last Name	First Name	Value					
Brüggemann	Jonas	5		5.00	OK		
Clicqué	Lennard	5					
Hasse	Lisabeth	5					
Heidkamp	Tessa	5					
Parameswaran	Birnavan	5		0	No size		
Takale	Aditi Vishwas	5		1	Trivial size		
				2	Small size		
				3	Medium size		
				5	Large size		
				8	Very large size		
				13	Too large (size)		
How to play planning poker							
1. Everyone type their number into their value field, don't hit return yet							
2. Someone, perhaps a product owner, count down 3.. 2.. 1..							
3. Then, everyone hit return to submit their value							

Sprint	Name	Description				
7		Complete	User Story	Notes	Acceptance Criteria	Definition of Done
	Unify requirements.txt files					
	Frontend: Display LLM result in Textbox					
	Onboard small model 1b					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki
	Prune models for only MIT license Models					
	Unify payload to Json					
	Module: Evaluate LLM performance					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki
	Module: Implement System property findings					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki
	Add multiple payload APIs					
8						
	Onboard and Test new LLM					
	Module: Add security analyzer module for generated code					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki

	Module: Add configurable timeouts for LLM generation					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki
	Module: Code Complexity: Complexity Measurement Tool (CMT)					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki
	Module: Code Complexity: CM					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki
	Frontend: make modules togglable in UI					
	Module: Detect redundant/duplicate test cases					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki
	Output exporter (as zip, json, http etc..)					
9						
	Research: test readability score (Flesch/Kincaid)					
	Module: Detect Edgecase accuracy of LLM research					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki
	Onboard and Test new LLM					

	Module: Map language to optimal model with fallback support					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki
	Module: Allow chaining outputs between models					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki
	Research: Befits of chaining methods					
	Use metrics to compare 1b, 3b, 7b and maybe 14b models					
	Research: How to measure code coverage with tools					
10						
	Onboard and Test new LLM					
	Module: Measure test readability score (Flesch/Kincaid)					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki
	Module: Check output for code coverage of the tests					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki
	Module: Flaky Test Identifier: Detect and flag non-deterministic/generated tests					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki

	Module: Show dependency hierarchy of code by logic flow					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki
	Try HPC to run the thing and test a larger model 50b+ parameters					
	Module: Auto-split large files into units for test generation					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki
	Research: how to train/enhance own LLM on HPC					
11						
	Onboard and Test new LLM					
	Use HPC to train our own LLM					
	Module: Auto-group similar test cases into suites					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki
	Clean the repo					

	Module: LLM testing of Code understandability, make LLM describe the code, let the same LLM (with no history) build the code from the description - > see if code passes the Unit tests.					# Definition of Done (DoD) Checklist - [] Code is tested - [] Code is reviewed by a peer - [] Code is merged into dev (without breaking) - [] Code is Documented in the Wiki
	Test current models for output consistency (deterministic)					
	Frontend: Diff viewer between original and tested source					
12						
	Research: different types of software test					
	CI: create Docker CI for running everything in CI					
	Research: can we run the models in the CI					
	Evaluate the performance of the models with and without the modules					
	Placeholder					
13						

	Ci: Create a CI flow that creates the uni tests and provides them as unit tests					
	Improve Usability of the frontend					
	Prepare Demo and create 3 Usecases					
	Clean the wiki					
	Overwork the Bill of Materials					
	Overwork the Architectuer Document					
	Refactor the Code					
	Placeholder					
14						
	Clean the repo					
	Finish documentation					
	Export Documentation (Wiki) as PDF					
	Create Demo Day slide					
	Create Demo Day video					
	Finalize user, (technical) design, and build/deploy documentation					