Hackfinity 3.0 Software Problem Statements

Problem Statement ID	Title of the Problem	Category	Theme	Problem Description
1. Machine Learning				
HF3-SW-01	**SBOM-Based Dependency Risk Detection System**	Software	Machine Learning	**Background:** Modern software projects depend heavily on open-source libraries, transitive packages, and third-party build components, making the software supply chain increasingly complex to secure. A single vulnerable dependency can expose an entire application, even when the main codebase appears safe. Existing tools often generate long vulnerability reports but fail to clearly prioritize risks from version changes, transitive dependencies, or delayed patch adoption across ecosystems like npm, Bun, and Python (pip/PyPI). **Description:** Develop an SBOM-based dependency risk detection system that analyzes a project's software bill of materials to identify vulnerable packages across package managers such as npm, Bun, and Python libraries. The system should parse dependency trees to detect transitive vulnerabilities, compare package versions against known CVE databases, and flag components with delayed or missing security patches. It should generate an automated risk score for each dependency based on severity, exploitability, and patch age, and present a prioritized vulnerability report highlighting the highest-risk components first. The tool should integrate into CI/CD pipelines to automatically scan SBOMs on each build, block merges when critical unpatched vulnerabilities are detected, and provide developers with clear remediation guidance. **Expected Solution:** An automated SBOM-based risk detection platform capable of scanning multi-language dependency trees (npm, Bun, Python, and similar ecosystems), identifying vulnerable and outdated packages, and assigning risk scores based on severity and exploitability. The solution should prioritize critical unpatched dependencies over low-risk findings, support seamless CI/CD integration for continuous scanning, and provide actionable remediation recommendations. By reducing alert fatigue and focusing attention on genuinely high-risk components, the system should strengthen software supply chain security and reduce the likelihood of exploitation through vulnerable dependencies.

Hackfinity 3.0 Software Problem Statements

HF3-SW-02	**Coverage-Guided Fuzz Harness Auto-Generator**	Software	Machine Learning	**Background:** Security testing of C/C++ libraries relies heavily on fuzzing to uncover memory-safety bugs, crashes, and undefined behaviour. However, building a working fuzz harness for an arbitrary library requires manual effort to understand the public API, infer valid input shapes, and construct correct setup/initialization sequences. Many libraries lack existing test harnesses, and developers often skip fuzzing entirely due to the time and expertise required, leaving vulnerabilities undiscovered until exploited in production. **Description:** Develop a coverage-guided fuzz harness auto-generator that analyzes an arbitrary C/C++ library's public API surface — function signatures, header files, and type definitions — using static analysis tools such as Clang AST/LibTooling to infer valid input shapes and call sequences. The system should detect functions requiring pointers, structs, or prior initialization (e.g., identifying that a function needs a context object from an init() call before use), and automatically generate a compilable libFuzzer or AFL++ harness. The generator should iteratively validate that the harness compiles, executes without immediate crashes from malformed setup, and measure coverage achieved, refining the harness across iterations to improve code path exploration. **Expected Solution:** An automated harness-generation pipeline that takes an unannotated C/C++ library as input and produces a working, compilable libFuzzer/AFL++ harness with measurable non-trivial coverage, demonstrated on 2–3 real-world or OSS-Fuzz-style targets (e.g., small parsing libraries). The solution should reduce the manual effort required to begin fuzzing a new codebase, correctly infer setup/initialization dependencies without human annotation, and report a clear coverage delta showing the harness exercises meaningful portions of the library's logic.
HF3-SW-03	**Secrets-Leak Scanner with Low False-Positive Semantic Validation**	Software	Machine Learning	**Background:** CI pipelines commonly use regex/entropy-based secrets scanners, which flood teams with false positives on test fixtures, UUIDs, and base64 blobs. This noise causes alert fatigue and adoption friction, while secrets committed and later "removed" often remain recoverable in git history, posing an ongoing security risk. **Description:** Develop a secrets-leak scanner that adds a semantic validation layer beyond regex/entropy detection, checking whether a flagged pattern matches a live, revocable API

				key format (format-only validation, never live-testing against endpoints). The tool should perform git-history-aware scanning across the full commit history, including squashed or rebased commits, to catch secrets that were committed and later deleted but still exist in history. It should build a low-noise classifier distinguishing real secrets from placeholders/test fixtures. **Expected Solution:** A CI-integrated secrets scanner with significantly lower false-positive rates than regex-only tools like Gitleaks/TruffleHog, capable of full git-history provenance tracking and semantic credential validation, improving real-world adoption and trust in automated secret detection.
HF3-SW-04	**Real-Time CI/CD Pipeline Poisoning Detector**	Software	Machine Learning	**Background:** CI/CD pipelines are increasingly targeted by supply-chain attacks (e.g., SolarWinds, Codecov-style compromises), where malicious actors inject unauthorized behaviour into build jobs. Existing tools detect issues post-hoc rather than during live pipeline execution, leaving a critical detection gap. **Description:** Build a tool that models a CI/CD pipeline's expected behaviour — which jobs should run, what artifacts they touch, what network egress is normal — as a baseline graph, then monitors live pipeline runs using sandboxed containers and sys call tracing (eBPF/strace) to detect anomalous deviations such as unexpected outbound connections, unauthorized file writes, or unexpected artifact uploads. **Expected Solution:** A real-time pipeline-poisoning detection system combining eBPF-level tracing with a lightweight anomaly-scoring model, flagging supply-chain compromise attempts in real time with a false-positive rate low enough for practical team adoption.
HF3-SW-05	**Self-Hosted Deployment Orchestration Platform**	Software	Machine Learning	**Background:** Small teams and educational projects increasingly seek self-hosted deployment solutions to avoid cloud vendor lock-in and reduce major cloud costs, while retaining full control over their database and application infrastructure. **Description:** Develop a self-hosted deployment orchestration platform with GitHub integration that triggers automated deployment checks on every push to a branch. On success, the deployment updates to the new commit; on failure, it automatically rolls back to the

				previous stable commit while maintaining detailed deployment logs. **Expected Solution:** A lightweight, self-hosted CI/CD-style orchestration platform providing automated build validation, safe rollback on failure, and transparent deployment logging — reducing reliance on costly managed cloud deployment services.
HF3-SW-06	**Custom Database Storage Engine with Crash-Consistent Durability Guarantees**	Software	Machine Learning	**Background:** Crash-consistency bugs in database storage engines rarely surface under normal testing, yet are a common source of real production data loss. Robust write-ahead logging and fsync ordering are notoriously difficult to get correct. **Description:** Build a minimal key-value storage engine (simplified LSM-tree or B-tree) with a write-ahead log, then systematically test crash consistency by injecting simulated crashes at every possible byte offset during writes, flushes, and compactions, verifying the database always recovers to a consistent state with no lost or corrupted committed writes. **Expected Solution:** A crash-consistent storage engine validated through systematic crash-point injection testing (similar to SQLite's crash-test harness or FoundationDB's deterministic simulation testing), demonstrating reliable recovery under all tested failure points.
HF3-SW-07	**Adverse-Selection-Aware Trade Execution Algorithm**	Software	Machine Learning	**Background:** Large trade orders risk significant market impact and adverse selection — the possibility that a counterparty has superior information. Existing execution strategies are either naive (TWAP/VWAP) or purely predictive without accounting for market impact, leaving a gap for smarter, risk-aware execution. **Description:** Build a smart order-routing/execution algorithm that predicts short-term adverse selection risk using order-flow imbalance and microstructure signals, dynamically switching between aggressive and passive execution strategies, combining classical market-microstructure theory (Almgren-Chriss) with a learned adverse-selection predictor. **Expected Solution:** A back tested execution algorithm that minimizes market impact and adverse selection risk more effectively than naive execution bots, validated against realistic slippage simulations.
HF3-SW-08	**Local Background Remover — Remove.bg**	Software	Machine Learning	**Background:** Commercial background-removal tools like Remove.bg charge per-image API

	/ Photo room Alternative			credits, creating cost barriers for bulk or offline use cases. **Description:** Build a fully local tool using an open-source segmentation model (rembg, U2-Net, or SAM) that removes backgrounds from images/video frames on the user's own GPU, with batch folder processing, edge refinement for difficult cases (hair, fur), and export as transparent PNG. **Expected Solution:** A local, cost-free background-removal tool with edge quality on difficult cases (hair, glass, motion blur) outperforming naive threshold-based cutout methods.
HF3-SW-09	**Urban Planning — Inferring Informal/Undocumented Road Networks from Satellite + Mobility Data**	Software	Machine Learning	**Background:** Informal settlements often have heavily used roads/paths absent from official map data. Satellite-only road extraction is a saturated problem, but fusing it with mobility data to resolve ambiguity remains underexplored, despite real applications in emergency response and utility planning. **Description:** Build a system that fuses satellite imagery (detecting path-like features via texture/edge analysis) with anonymized mobility trace data (GPS pings) to infer and generate an accurate, previously undocumented road graph, resolving ambiguous cases (e.g., distinguishing a dry riverbed from an actual path) using corroborating mobility evidence. **Expected Solution:** A multi-modal data-fusion system producing accurate undocumented road network maps for informal settlements, supporting emergency response and infrastructure planning in areas ignored by official maps.
2. NLP & Conversational AI				
HF3-SW-10	**Handwritten Answer-Sheet Auto-Grader for Subjective/Descriptive Answers**	Software	NLP and Conversational AI	**Background:** Manually grading handwritten subjective answer sheets is time-consuming and inconsistent, especially with mixed print/cursive handwriting, varying pen quality, crossed-out text, and diagrams embedded in prose. Automated grading tools often fail silently on low-quality OCR, producing unreliable grades. **Description:** Build a grading pipeline that processes scanned handwritten answer sheets — including mixed-script answers (English/Hindi/regional languages), embedded diagrams, and margin overflow — and produces a rubric-aligned grade with justification. The system should flag low-confidence

				transcriptions for human review rather than silently mis-scoring. **Expected Solution:** A confidence-aware, rubric-based auto-grading system for descriptive answers that gracefully handles messy real-world handwriting and OCR failure, escalating uncertain cases to human graders instead of hallucinating scores.
HF3-SW-11	**Legal Tech — Contradiction Detection Across a Contract's Cross-References**	Software	NLP and Conversational AI	**Background:** Long legal contracts contain nested defined terms, cross-references, and amendments that can silently modify earlier clauses, creating logical contradictions that are difficult to detect manually. Existing contract-review tools invest heavily in cross-reference resolution, which most teams underestimate. **Description:** Build a tool that parses a long legal contract with nested defined terms and cross-references, resolves references correctly (e.g., "as defined in Section 4.2"), constructs a structured logical representation of obligations and conditions, and detects contradictions such as an amendment silently changing a payment term without acknowledgment elsewhere. **Expected Solution:** A contract-analysis tool capable of accurate cross-reference resolution and logical contradiction detection, surfacing conflicts (e.g., Net-30 vs. Net-45 payment terms) that manual review often misses.
HF3-SW-12	**Grounded Dialogue Summarization That Preserves Speaker Attribution and Commitment Tracking**	Software	NLP and Conversational AI	**Background:** Standard abstractive summarization models often blur speaker attribution and fail to distinguish firm commitments from tentative statements, since most training data doesn't explicitly supervise for these distinctions — a major gap for meeting/conversation summarization tools. **Description:** Build a meeting/conversation summarizer that accurately tracks who said what, distinguishes firm commitments from hedged statements, and produces summaries where every action item is traceable to a specific speaker and turn, evaluated on messy multi-speaker transcripts with overlapping speech and ambiguous pronouns. **Expected Solution:** A dialogue summarization system with reliable speaker-attribution and commitment-tracking accuracy, evaluated using a custom rubric beyond ROUGE, producing trustworthy, traceable action-item summaries from real-world conversations.

Hackfinity 3.0 Software Problem Statements

HF3-SW-13	**Disaster Response — Cross-Language Rumour/Misinformation Divergence Tracking During Crises**	Software	NLP and Conversational AI	**Background:** Standard misinformation detection treats posts independently and monolingually, missing how rumours mutate as they spread across language communities during crises — an underexplored problem relevant to disaster-response coordination. **Description:** Build a system that tracks how the same rumour or claim mutates as it spreads across different language communities on social media, clustering semantically equivalent claims across languages and building a "mutation graph" showing how facts were exaggerated, altered, or fabricated at each retelling. **Expected Solution:** A cross-lingual claim-lineage tracking system that visualizes rumour mutation graphs across language/community boundaries, helping disaster-response teams identify dangerously amplifying misinformation in real time.
HF3-SW-14	**Multilingual Low-Resource Intent + Slot Learner**	Software	NLP and Conversational AI	**Background:** Intent classification and slot-filling systems typically require large labelled datasets, making them impractical for low-resource languages. Cross-lingual transfer and meta-learning offer a path to generalization without extensive per-language data. **Description:** Build an end-to-end intent classification and slot-filling system that generalizes to unseen low-resource languages via cross-lingual transfer and meta-learning, addressing domain adaptation, few-shot slot labelling, script alignment, and code-switching, using Py Torch, Hugging Face Transformers, XLM-R, MAML/PEFT, fair seq, and Sentence Piece. **Expected Solution:** A multilingual intent/slot-filling model achieving strong F1 (slots) and accuracy (intent) on held-out low-resource languages, demonstrating effective cross-lingual generalization.
HF3-SW-15	**Robust Dialogue State Tracker under Noisy ASR**	Software	NLP and Conversational AI	**Background:** Task-oriented dialogue systems must track dialogue state accurately even when input comes from noisy speech recognition (ASR errors, disfluencies), which existing dialogue state trackers often handle poorly. **Description:** Build a dialogue state tracker for multi-domain task-oriented dialogue that models ASR uncertainty, supports incremental state updates, and operates within latency constraints, using Rasa/ConvLab-3, Kaldi/Whisper, BERT-based DST, and lattice/confusion network processing.

				Expected Solution: A dialogue state tracking system achieving strong joint goal accuracy even with simulated noisy ASR input, demonstrating robustness for real-world voice-driven task-oriented applications.
HF3-SW-16	**Air Canvas — Gesture-Controlled Virtual Whiteboard**	Software	NLP and Conversational AI	**Background:** Touchless, gesture-based interaction is increasingly relevant for accessibility and natural interfaces, yet most drawing/annotation tools still depend on mouse, touchscreen, or stylus input. **Description:** Build a real-time application where hand gestures captured via webcam control drawing/annotation on a virtual canvas — tracking hand/fingertip movement live, rendering ink, and supporting color/thickness selection, stroke-specific erasing, canvas clearing, and export to image/PDF, using MediaPipe Hands or OpenCV-based hand-landmark detection. **Expected Solution:** A responsive gesture-controlled whiteboard application with smooth fingertip tracking, reliable gesture disambiguation (e.g., fist = erase, open palm = clear, pinch = select color), and exportable output — demonstrable live with no physical input device. Add an AI tutor or voice assistant.
3. Generative AI & LLM Applications				
HF3-SW-17	**Recovery of Deleted Data and Associated Metadata from XFS and Btrfs Filesystems**	Software	Generative AI and LLM	**Background:** Modern filesystems like XFS and Btrfs use journaling and copy-on-write mechanisms that make deleted-file recovery significantly harder than on traditional filesystems, complicating digital forensic investigations that depend on reconstructing timelines and preserving evidence integrity. **Description:** Develop a forensic data recovery solution capable of recovering deleted files from XFS and Btrfs filesystems while preserving metadata. The tool should support multiple file formats (documents, images, archives, logs, executables, scripts, databases) and provide a GUI or CLI for browsing recovered data and generating forensic automated reports. **Expected Solution:** A reliable forensic recovery tool for XFS/Btrfs filesystems that recovers deleted files with full metadata integrity, supports diverse file types, and produces audit-ready forensic reports for investigative use, and report generation must be automated.
HF3-SW-18	**Detecting LLM-Generated Text That**	Software	Generative AI and LLM	**Background:** AI-generated-text detectors like GPT Zero typically report strong accuracy on

	Has Been Adversarially Paraphrased to Evade Detection			raw LLM output but are rarely tested against paraphrasing attacks (e.g., back-translation, secondary paraphrasing models), which represent the real-world evasion case relevant to academic integrity and misinformation detection. **Description:** Build an AI-generated-text detector that remains robust against paraphrasing-based evasion attacks, and transparently report the detector's performance degradation curve as paraphrasing intensity increases, rather than only reporting clean-output accuracy. **Expected Solution:** A paraphrase-robust AI-text detector with a clearly documented degradation profile under adversarial paraphrasing, offering a more credible and transparent alternative to existing detectors that break down under real-world evasion attempts.
HF3-SW-19	**The Gamified Narrative-Driven LMS (The Curriculum RPG)**	Software	Generative AI and LLM	**Background:** Standard LMS platforms are structured like static file cabinets — folders, PDFs, drop boxes — which kills curiosity and relies entirely on learner discipline to complete courses. **Description:** Design an LMS that transforms a standard course syllabus into a dynamic, narrative-driven RPG. Students unlock story chapters, earn skill points, and level up a character profile as they complete modules, with the ability to choose personalized learning paths (e.g., coding-heavy vs. theory-heavy) toward the same learning objective. **Expected Solution:** An engaging, gamified LMS platform (React/Phaser.js, Node.js, PostgreSQL) that increases learner motivation and completion rates through narrative progression, character customization, and flexible learning paths.
HF3-SW-20	**Deterministic Replay Debugger for Multithreaded Programs**	Software	Generative AI and LLM	**Background:** Race-condition bugs in multithreaded programs often manifest rarely (e.g., 1-in-10,000 runs), making them extremely difficult to reproduce and debug. Existing replay tools (rr, Chronon) require significant engineering to capture sufficient execution detail without altering timing. **Description:** Build a tool that records execution of a multithreaded program (thread scheduling, memory accesses, sys call ordering) with low enough overhead to avoid disturbing the timing being captured, then deterministically replays that exact execution for step-through debugging.

				Expected Solution: A scoped-down deterministic replay debugger capable of reliably reproducing a specific class of race-condition bugs (e.g., data races) on a toy or real program, enabling developers to step through and inspect state at the exact moment of failure Explain crashes, suggest fixes, summarize replay logs, assist developers.
HF3-SW-21	**Photorealistic 3D Asset Generator with Editable Parameters**	Software	Generative AI and LLM	**Background:** The growth of gaming, VR, digital product design, and simulation has increased demand for high-quality, quickly created 3D assets. Generating textured, rigg able 3D models from a single image or text prompt remains difficult, as it requires accurate geometry inference, realistic textures, multi-view consistency, and editable output. **Description:** Develop a photorealistic 3D asset generator that creates editable, textured, and rigg able 3D models from limited input (image or text prompt), supporting geometry inference, texture mapping, multi-view consistency, and stable UV maps, with seamless integration into modelling tools such as Blender. **Expected Solution:** A generative 3D asset pipeline producing production-ready, editable meshes with realistic textures and reliable compatibility with standard 3D software, reducing manual 3D modelling effort for games, VR, and simulation applications.
HF3-SW-22	**Hallucination Hunter**	Software	Generative AI and LLM	**Background:** LLMs frequently sound confident even when factually incorrect, undermining user trust. Existing tools rarely break down generated answers into individually verifiable claims with transparent sourcing. **Description:** Build a system that takes any LLM-generated answer, splits it into individual factual claims, checks each claim against a trusted source, and displays which parts are verified, unverifiable, or incorrect, along with the source used for each check — using prompting for claim extraction, retrieval/grounding, and confidence scoring. **Expected Solution:** A claim-level fact-verification layer for LLM outputs that visually distinguishes verified, unverifiable, and false claims with source attribution, improving user trust and transparency in AI-generated answers.
HF3-SW-23	**Offline AI Companion — Zero-API, Fully Local**	Software	Generative AI and LLM	**Background:** Most AI assistants depend on cloud API calls, creating internet dependency, privacy concerns, and latency issues. There is growing demand for fully local AI assistants

				that run entirely on-device using open-source models. **Description:** Build an AI assistant that runs completely on local hardware — no external API calls at inference time — using open-source LLMs (e.g., Llama 3.2, Phi-3, Mistral, Gemma) with quantization (GGUF/AWQ/GPTQ) tuned to available GPU/VRAM. The system should support a real use case (document Q&A with local RAG, coding helper, or voice assistant with local Whisper + local TTS) and display live proof of local execution (tokens/sec, GPU usage). **Expected Solution:** A fully offline, privacy-preserving AI assistant demonstrating real-world utility with measurable local inference performance, verified functional with no internet connectivity.
HF3-SW-24	**Retrieval-Augmented Conversational QA with Source Attribution**	Software	Generative AI and LLM	**Background:** LLM-based QA systems are prone to hallucination and often fail to provide precise, verifiable source citations, limiting their trustworthiness for factual queries. **Description:** Build a RAG-based conversational agent that answers user queries and returns precise source citations with extracted evidence spans, addressing retrieval precision, citation alignment, and hallucination control, using Faiss/Elastic Search, DPR/MiniLM embeddings, and an LLM generator with citation templates. **Expected Solution:** A retrieval-augmented QA system achieving strong answer F1 and citation precision, providing users with verifiable, source-attributed answers rather than unverified generated text.
4. Open Innovation				
HF3-SW-25	**Open Innovation**	Software	Open Innovation	----------