UNIVERSITETI I PRISHTINËS
"HASAN PRISHTINA"
UNIVERSITY OF PRISHTINA

Rr.Xhorxh Bush, Ndërtesa e Rektoratit, 10 000 Prishtinë,Republika e Kosovës
Tel: +383 38 244 183 • E-mail: rektorati@uni-pr.edu • www.uni-pr.edu

Fakulteti i Inxhinierisë Elektrike dhe Kompjuterike
Faculty of Electrical and Computer Engineering

Abstrakt i zgjeruar i punimit të diplomës
Niveli Master

Kandidatja	Titulli i temës
Erëblina Berisha	Zhvillimi i një sistemi Multi-Agjent për rishikimin automatik të kodit të bazuar në LLM me mbështetje shumëgjuhëshe

1. Hyrje dhe kontekst

Zhvillimi modern i softuerit kërkon standarde të larta të cilësisë, sigurisë dhe mirëmbajtshmërisë së kodit burimor. Me rritjen e kompleksitetit të aplikacioneve dhe përdorimin e teknologjive të ndryshme programuese, procesi i rishikimit manual të kodit është bërë gjithnjë e më i vështirë, më i kushtueshëm dhe i prirur ndaj gabimeve njerëzore. Për këtë arsye, organizatat dhe ekipet e zhvillimit po përdorin gjithnjë e më shumë mjete të automatizuara për analizën dhe vlerësimin e kodit.

Përparimet në Inteligjencën Artificiale, veçanërisht në fushën e Modeleve të Mëdha Gjuhësore (LLM), kanë sjellë mundësi të reja për analizën e kodit burimor. Këto modele janë të afta të kuptojnë strukturën dhe semantikën e kodit, të identifikojnë probleme potenciale dhe të ofrojnë rekomandime në gjuhë natyrore. Megjithatë, përdorimi i vetëm i LLM-ve nuk garanton gjithmonë saktësi të lartë teknike. Nga ana tjetër, mjetet tradicionale të analizës statike janë shumë të sakta në identifikimin e problemeve të bazuara në rregulla, por nuk ofrojnë interpretim të avancuar kontekstual.

Në këtë kontekst, ky punim trajton zhvillimin e një sistemi multi-agjent për rishikimin automatik të kodit, i cili kombinon analizën statike me aftësitë interpretuese të inteligjencës artificiale për të krijuar një zgjidhje më të saktë, më të besueshme dhe më të dobishme për zhvilluesit e softuerit.

2. Problematika dhe motivimi

Rishikimi i kodit është një proces thelbësor për sigurimin e cilësisë dhe sigurisë së aplikacioneve softuerike. Megjithatë, në projekte të mëdha dhe shumëgjuhëshe, shqyrtimi manual i kodit kërkon kohë të konsiderueshme dhe shpesh nuk arrin të identifikojë të gjitha problemet ekzistuese. Kjo paraqet sfida të veçanta në identifikimin e dobësive të sigurisë, gabimeve logjike dhe praktikave jo të mira të programimit.

Mjetet tradicionale të analizës statike, si SonarQube, PMD apo ESLint, janë efektive në identifikimin e gabimeve teknike dhe code smells, por zakonisht nuk ofrojnë shpjegime të

kuptueshme ose rekomandime të personalizuara për zhvilluesit. Në anën tjetër, modelet e mëdha gjuhësore ofrojnë aftësi të avancuara interpretimi, por mund të prodhojnë rezultate të pasakta ose jo konsistente.

Motivimi kryesor i këtij punimi është krijimi i një qasjeje hibride që bashkon avantazhet e analizës statike me fuqinë interpretuese të inteligjencës artificiale. Gjithashtu, literatura ekzistuese tregon mungesë të zgjidhjeve që kombinojnë analiza statike, LLM dhe arkitektura multi-agjent në një sistem të vetëm me mbështetje shumëgjuhëshe. Kjo krijon nevojën për zhvillimin e një sistemi të ri që mund të përmirësojë efikasitetin dhe saktësinë e procesit të rishikimit të kodit.

3. Qëllimi dhe objekti i punimit

Qëllimi kryesor i punimit është zhvillimi i një sistemi multi-agjent për rishikimin automatik të kodit burimor, i bazuar në kombinimin e analizës statike dhe modeleve të mëdha gjuhësore, me mbështetje për disa gjuhë programuese.

Objekti i punimit përfshin:

- Analizën e literaturës ekzistuese lidhur me rishikimin automatik të kodit, sistemet multi-agjent dhe modelet e mëdha gjuhësore.
- Projektimin e një arkitekturë multi-agjent për analizën e kodit.
- Implementimin e agjentëve të specializuar për analizën e kodit, kontrollin e sigurisë dhe vlerësimin e cilësisë.
- Integrimin e teknikave për analizë statike me aftësitë interpretuese të një modeli LLM.
- Sigurimi i mbështetjes për gjuhë të ndryshme programuese.
- Zhvillimi i një mekanizmi për gjenerimin e raporteve automatike të analizës.
- Integrimi i sistemit me platforma të kontrollit të versioneve për të mundësuar rishikimin automatik të kodit gjatë proceseve të Continuous Integration dhe Continuous Deployment (CI/CD).
- Vlerësimi i performancës dhe efektivitetit të sistemit përmes skenarëve të testimit dhe metrikave përkatëse.

4. Përmbajtja teorike

Pjesa teorike e punimit mbështetet në disa fusha kryesore të inxhinierisë softuerike dhe inteligjencës artificiale. Fillimisht trajtohet koncepti i rishikimit të kodit si një mekanizëm për sigurimin e cilësisë së softuerit dhe reduktimin e defekteve gjatë zhvillimit. Gjithashtu analizohet roli i analizës statike të kodit, e cila mundëson identifikimin e gabimeve, code smells dhe dobësive të sigurisë pa ekzekutuar programin.

Një pjesë e rëndësishme i kushtohet sigurisë së kodit dhe dobësive më të zakonshme si SQL Injection, Cross-Site Scripting (XSS), Hardcoded Credentials dhe Insecure Deserialization. Punimi mbështetet në standarde të njohura si OWASP Top 10 dhe CWE për klasifikimin e këtyre dobësive.

Në vazhdim paraqitet përdorimi i inteligjencës artificiale në inxhinierinë softuerike, me fokus të veçantë në Modelet e Mëdha Gjuhësore (LLM). Shpjegohen karakteristikat e arkitekturës Transformer, koncepti i prompt engineering dhe përdorimi i modeleve moderne si GPT, Llama, Claude dhe Gemma.

Gjithashtu trajtohet teoria e sistemeve multi-agjent, ku disa agjentë autonomë bashkëpunojnë për zgjidhjen e problemeve komplekse përmes koordinimit, memories së përbashkët dhe

arsyetimit të shpërndarë. Kjo teori përbën bazën konceptuale të arkitekturës së propozuar në këtë punim.

5. Metodologjia dhe implementimi praktik

Metodologjia e ndjekur për realizimin e punimit përfshin katër faza: analizën e literaturës, projektimin e sistemit, implementimin dhe vlerësimin eksperimental. Pas studimit të literaturës shkencore dhe mjeteve ekzistuese, u projektua një arkitekturë hibride multi-agjent që kombinon analizën statike me inteligjencën artificiale. Sistemi u implementua në Python duke përdorur Flask për aplikacionin web dhe Google GenAI SDK për integrimin me modelin Gemma 4.

Arkitektura përbëhet nga një orkestrues qendror, memoria e përbashkët dhe tre agjentë kryesorë: Code Analysis Agent, Security Analysis Agent dhe Quality Review Agent.

Code Analysis Agent realizon validimin sintaksor, analizën AST, llogaritjen e metrikave dhe identifikimin e code smells. Security Analysis Agent identifikon dobësi të sigurisë si SQL Injection dhe përdorimin e kredencialeve të ngulitura në kod. Quality Review Agent vlerëson cilësinë dhe mirëmbajtshmërinë e kodit.

Për të siguruar mbështetje shumëgjuhëshe, sistemi integron parserë të ndryshëm për Python, Java, JavaScript dhe TypeScript. Përveç analizës përmes ndërfaqes web, sistemi është integruar me GitHub Actions për analizën automatike të Pull Request-eve në procese CI/CD.

6. Rezultatet dhe diskutimi

Për vlerësimin e sistemit u zhvillua një kornizë automatike testimi e bazuar në një dataset Ground Truth me raste testuese të etiketuara manualisht. Testimi përfshiu vlerësimin funksional, testimin e integrimit dhe analizën e performancës së sistemit.

Rezultatet eksperimentale treguan performancë të lartë në identifikimin e problemeve reale në kod. Sistemi arriti Precision prej 0.842, Recall prej 1.000 dhe F1 Score prej 0.909. Këto rezultate tregojnë se sistemi është në gjendje të identifikojë pothuajse të gjitha problemet ekzistuese duke minimizuar numrin e problemeve të humbura.

Analiza tregoi se kombinimi i analizës statike me modelin Gemma 4 përmirëson ndjeshëm interpretimin dhe kuptueshmërinë e rezultateve krahasuar me mjetet tradicionale. Arkitektura multi-agjent mundësoi specializimin e analizave dhe ndarjen e përgjegjësive ndërmjet komponentëve, duke rritur modularitetin dhe shkallëzueshmërinë e sistemit.

Megjithatë, u identifikuan edhe kufizime të caktuara, si varësia nga cilësia e përgjigjeve të modelit gjuhësor dhe kufizimet që lidhen me analizën e projekteve shumë të mëdha.

7. Përfundime

Ky punim prezantoi zhvillimin e një sistemi multi-agjent për rishikimin automatik të kodit burimor, i cili kombinon analizën statike me aftësitë e modeleve të mëdha gjuhësore. Arkitektura e propozuar përfshin agjentë të specializuar për analizën e kodit, sigurinë dhe cilësinë, të koordinuar përmes një mekanizmi orkestrimi dhe memories së përbashkët.

Rezultatet e vlerësimit treguan se qasja hibride ofron performancë të lartë në identifikimin e problemeve të kodit dhe në gjenerimin e rekomandimeve të kuptueshme për zhvilluesit. Integrimi me GitHub Actions demonstroi gjithashtu potencialin praktik të sistemit në procese moderne të zhvillimit softuerik.

Kontributi kryesor i punimit qëndron në kombinimin e analizës statike, inteligjencës artificiale dhe arkitekturës multi-agjent në një sistem të vetëm me mbështetje shumëgjuhëshe. Rezultatet

tregojnë se kjo qasje përfaqëson një drejtim premtues për zhvillimin e mjeteve inteligjente të sigurimit të cilësisë së softuerit dhe krijon bazë për zgjerime dhe hulumtime të ardhshme në këtë fushë.