



# 프로젝트 기획서

## 🚀 프로젝트명: AI 기반 코드 보안 분석기 개발 (AI-Powered Code Security Analyzer Development)

- 목적: 소프트웨어 소스 코드에서 보안 취약점을 자동으로 탐지하고, 적절한 보안 권장 사항을 제시하는 AI 기반 서비스 구축
- 핵심기술: NLP 기반 코드 분석 (GPT-3.5 API, CodeBERT)  
+ 보안 데이터셋 활용 (OWASP, CWE, CVE)
- 참여인원: 1인(소프트웨어학과 4학년 2022564004 김다빈)
- 개발기간: 2025년 3월 5일 ~ 2025년 4월 29일

## 1 프로젝트 개요

### ✅ 배경 및 필요성

- 코드의 보안 취약점을 발견하는 것이 쉽지 않음
- OWASP Top 10, CWE를 학습한 AI 모델을 통해 **안전성이 보장된 코드 작성 가능**
- AI 모델을 활용하면 **자동으로 코드의 보안 취약점 분석**이 가능하고, **예기치 못한 사고를 예방 가능**

### ✅ 주요 기능

- ◆ **취약점 자동 탐지**: 코드에서 SQL Injection, XSS, Buffer Overflow 등 주요 취약점 감지
- ◆ **취약점 설명 & 해결 방안 추천**: GPT-3.5-API, CodeBERT를 활용하여 코드 리뷰 및 보안 권장 사항 제공
- ◆ **REST API 제공**: 웹사이트, VScode(IDE 플러그인), CLI 등 연동 가능

## 2 개발 계획

### ✅ 📌 시스템 구조

[사용자 코드 입력 → AI 모델 분석 (취약점 탐지) → 취약점 보고서 제공] 의 단계로 구성

### ✅ 📌 기술 스택

영역	기술
AI 모델	CodeBERT, GPT-3.5-API
보안 데이터	OWASP, CWE, CVE 등 보안 취약점 데이터셋 활용
백엔드	FastAPI 기반의 REST API
프론트엔드	GitHub 웹호스팅 활용(GitHub Pages)
배포 환경	Render

### 3 AI 모델 설계

#### ✓ AI 모델 개요

- **입력:** 사용자의 소스 코드 (Python, C)
- **처리:** CodeBERT + GPT-3.5-API로 취약점 탐지 및 분석
- **출력:** 취약점이 포함된 코드 라인, 취약점 설명, 보안 권장 사항

#### ✓ AI 모델 학습 데이터

##### ◆ 기본 학습 데이터

- OWASP Top 10 보안 취약점 사례
- CWE 데이터셋 (Common Weakness Enumeration)
- CVE 데이터 (실제 해킹 사례 기반 취약점)
- CodeXGLUE 깃허브 데이터셋(trains)

##### ◆ Fine-Tuning 과정

- CodeBERT 모델을 활용하여 코드 취약점 감지 모델 학습
- GPT-3.5-API를 활용하여 보안 권장 사항 자동 생성

### 4 기능 상세

#### ✓ 기능 1: 코드 취약점 자동 분석

- 입력된 코드에서 보안 취약점이 있는 부분을 탐지
- ex) 예제 (Python 코드에서 SQL Injection 감지)

```
user_input = request.GET.get('id')
query = "SELECT * FROM users WHERE id = " + user_input # 🔥 취약점
```

```
{
  "prediction": "⚠️ SQL Injection 취약점 감지",
  "label": "SQL_Injection",
  "security score": 30,
  "report": "🔴 **1. 취약점 설명**\n해당 코드에서 발견된 취약점은 ...",
  "model_reference": "Use parameterized queries instead of string concatenation."
}
```

## ✅ 기능 2: 보안 점수 산출

- 코드 내 취약점 종류에 따라 라벨, 보안 점수, 메시지 할당

취약점 종류	보안 점수	메시지
SQL_Injection	30	⚠️ SQL Injection 취약점 감지
Hardcoded_Password	30	⚠️ Hardcoded Password 감지
XSS	30	⚠️ XSS 취약점 감지
Safe_Code	100	✅ 완전한 방어 코드 (Safe Code)
Other	70	⚠️ 잠재적 취약점이 있는 코드 (Other)

🔍 보안 분석 결과:

```
{
  "prediction": "SQL_Injection"
  "security_score": 60,
  "rating": "Medium ⚠️ (주의)"
}
```

## ✅ 기능 3: REST API 제공

- API를 통해 웹, IDE, 터미널에서 활용 가능
- 예제 API 요청

```
POST /analyze
Content-Type: application/json
{
  "code": "user_input = request.GET.get('id')\nquery = 'SELECT * FROM users WHERE id = ' + use
r_input"
}
```

```
{
  "prediction": "⚠️ SQL Injection 취약점 감지",
  "label": "SQL_Injection",
  "security score": 30,
  "report": "🔴 **1. 취약점 설명**\n해당 코드에서 발견된 취약점은 ...",
  "model_reference": "Use parameterized queries instead of string concatenation."
}
```

## 5 일정 계획 (캡스톤디자인 진행 계획)

기간	목표
1주차(3/5 ~ 3/11)	프로젝트 기획 & 보안 데이터셋 수집
2주차(3/12 ~ 3/18)	CodeBERT 모델 Fine-tuning, 데이터셋 전처리
3주차(3/19 ~ 3/25)	모델 학습
4주차(3/26 ~ 4/1)	GPT-4 API 연동하여 취약점 보고서 생성 서비스 구현
5주차(4/2 ~ 4/8)	FastAPI로 REST API 구현(1)
6주차(4/9 ~ 4/15)	FastAPI로 REST API 구현(2)
7주차(4/16 ~ 4/22)	API 배포 (Render) 및 테스트
8주차(4/23 ~ 4/29)	CLI 패키지, 웹사이트 등 결과물 활용 자료 준비

## 6 기대 효과

- ✓ 개발자의 보안 수준 향상 & 코드 품질 개선
- ✓ 사용자 코드 입력만으로 간편하게 API로 취약점 보고서 받아볼 수 있음
- ✓ 보안 취약점 보고서를 활용해 취약점으로 인한 사고 예방 가능
- ✓ API를 활용해 CLI, 웹사이트, IDE 플러그인 등 다양한 환경에서 간편하게 취약점 점검 가능
- ✓ 기존 시스템에 API를 추가해 AI 기반 자동 취약점 분석 시스템 구축 가능
- ✓ 시스템 운용 비용/인력 절감 & 유지보수 효율성 증가