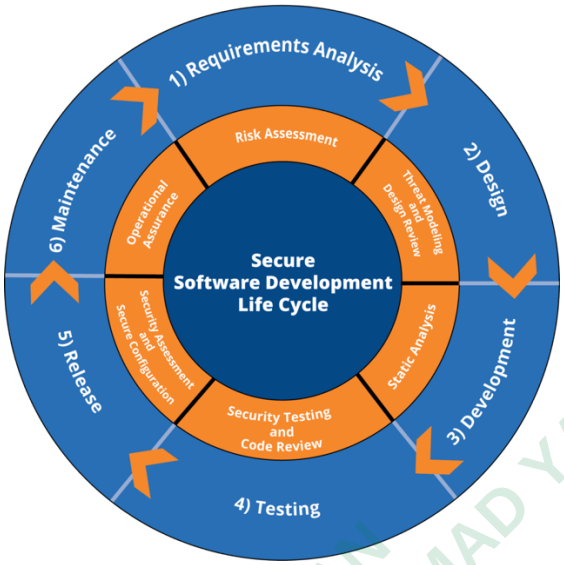
BAB 3
METODE PENELITIAN

Metode Penelitian, dijelaskan mengenai pendekatan dan tahapan yang digunakan dalam penelitian rancang-bangun ini, yang bertujuan untuk mereduksi atau mengeliminasi permasalahan keamanan. Metode utama yang diterapkan adalah *Secure Software Development Life Cycle* (SSDLC) dengan fokus pada *Static Application Security Testing* (SAST) untuk mendeteksi kerentanan sejak dini. Bab ini merinci tahapan SSDLC, yaitu Perencanaan, Perancangan, Pengembangan, Pengujian, serta Rilis dan Pemeliharaan. Dijelaskan pula bahan penelitian berupa repositori GitHub, serta alat penelitian yang mencakup perangkat keras (laptop dan tiga *Virtual Private Server* untuk Jenkins, SonarQube, Docker) dan perangkat lunak (VSCode, GitHub, SonarQube, Jenkins, Docker, dan Reverse Proxy). Terakhir, bab ini memaparkan jalan penelitian yang terstruktur berdasarkan siklus pengembangan *software* dengan pendekatan SSDLC dan penerapan SAST.

3.1 Secure Software Development Life Cycle (SSDLC)

Metode penelitian ini menggunakan pendekatan *Secure Software Development Life Cycle* (SSDLC) untuk memastikan bahwa aspek keamanan menjadi bagian integral dalam setiap tahap pengembangan *software*. *Static Application Security Testing* (SAST) diterapkan dalam proses ini guna mendeteksi potensi kerentanan sejak dini, sebelum aplikasi dirilis ke lingkungan produksi, Seperti pada **Gambar 3.1**.

PERPUSTAKAAN UNIVERSITAS JENDERAL ACHMAD YANI YOGYAKARTA



Gambar 3.1 Metode SSDLC (https://www.digitalmaelstrom.net/)

3.1.1 *Planning* (Perencanaan)

Penelitian dimulai dengan Identifikasi kebutuhan proyek yang akan dikembangkan. Menentukan *framework* dan teknologi yang digunakan dalam pengembangan *software*. Menetapkan kebijakan keamanan yang akan diterapkan sejak awal, termasuk penerapan SAST dengan SonarQube dalam *pipeline* CI/CD.

3.1.2 *Design* (Perancangan)

Selanjutnya dilakukan *Threat Modelling*, yaitu analisis terhadap potensi ancaman keamanan yang dapat terjadi dalam sistem dengan menentukan arsitektur sistem yang aman, termasuk metode autentikasi, enkripsi, dan proteksi data *sensitive*, dengan merancang struktur kode yang sesuai dengan standar *secure coding* untuk mengurangi risiko keamanan.

3.1.3 *Development* (Pengembangan)

Fase pengembangan diikuti dengan prinsip *secure coding* di mana setiap *commit* ke *repository* GitHub akan secara otomatis memicu *pipeline* Jenkins untuk melakukan analisis keamanan kode menggunakan SonarQube (SAST). Jika

PERPUSTAKAAN UNIVERSITAS JENDERAL ACHMAD YANI YOGYAKARTA

ditemukan celah keamanan, proses *build* akan dihentikan, dan pengembang harus memperbaiki kode sebelum melanjutkan ke tahap selanjutnya.

3.1.4 *Testing* (Pengujian)

Fase pengujian adalah melakukan *Integrated Testing*, yang mencakup pengujian fungsionalitas, performa, dan keamanan sistem, hasil analisis SAST dari SonarQube akan diperiksa untuk dipastikan tidak adanya celah keamanan sebelum aplikasi di *deploy*.

3.1.5 *Release* (Rilis dan Pemeliharaan)

Fase terakhir setelah aplikasi dipastikan aman, dilakukan *deployment* dan evaluasi menggunakan Docker ke server, dipastikan bahwa *source code* pada aplikasi web tidak adanya kerentanan.

3.2 BAHAN PENELITIAN

Bahan penelitian utama yang digunakan dalam studi ini adalah sebuah repositori dari platform GitHub. Repositori tersebut dapat diakses melalui tautan https://github.com/0x1Jar/DevSecOps-Demo-TA. Nama dari repositori ini adalah DevSecOps-Demo-TA. Fungsi utama dari repositori ini nantinya adalah sebagai tempat penyimpanan kode sumber (*source code*). Repositori ini juga akan digunakan untuk mengidentifikasi potensi kerentanan keamanan pada aplikasi web. Kode yang tersimpan dalam repositori ini secara spesifik digunakan sebagai basis pengembangan aplikasi yang selanjutnya akan diuji menggunakan metode *Static Application Security Testing* (SAST).

Berdasarkan struktur *file* yang terdapat dalam proyek ini, beragam tipe file utama dapat diidentifikasi dengan kemungkinan fungsi yang berbeda. Untuk bagian *Frontend Web*, proyek ini mencakup file .html seperti index.html dan vulnerable.html yang berfungsi sebagai kerangka utama halaman web. Pengaturan tampilan dan gaya (*styling*) halaman web diatur oleh file .css, .sass, dan .scss yang berlokasi di dalam direktori assets/css/. Sementara itu, untuk menambahkan

UNIVERSITAS JENDRAL ACHMAD YANI YOGYAKARTA PERPUSTAKAAN

interaktivitas dan fungsionalitas dinamis pada sisi klien (*browser*), digunakan file .js yang berada di direktori assets/js/ dan js/.

Pada segmen *Backend & Server*, kode sisi server yang ditulis dalam Node.js ditemukan dalam file .js di direktori server/, dengan contoh seperti vulnerable-server.js yang kemungkinan besar merupakan aplikasi web server. Skrip Python, kemungkinan berisi logika *backend* atau contoh kerentanan (vulnerable.py), terletak di direktori python/. Selain itu, terdapat pula file nginx.conf yang merupakan file konfigurasi untuk Nginx, sebuah *web server* atau *reverse proxy*.

Bagian *Database* proyek ini mencakup file .sql yang berada di direktori sql/. File-file ini kemungkinan besar berisi perintah-perintah SQL untuk mendefinisikan struktur *database* atau memuat data sampel, seperti yang terlihat pada vulnerable.sql.

Dalam konteks *DevOps & Automasi*, proyek ini menggunakan Dockerfile dan .dockerignore. File-file ini berfungsi untuk membuat *container* Docker, yang bertugas mengemas aplikasi beserta lingkungannya agar mudah didistribusikan dan dijalankan. Selain itu, Jenkinsfile juga disertakan sebagai skrip untuk Jenkins, yang merupakan *automation server*. Jenkinsfile ini mendefinisikan alur CI/CD (Continuous Integration/Continuous Deployment) proyek. File konfigurasi untuk proyek Node.js adalah package.json, yang isinya mendefinisikan *dependencies* (pustaka yang digunakan) dan skrip-skrip (seperti untuk minifikasi CSS/JS).

Untuk *Dokumentasi & Konfigurasi Lainnya*, proyek ini memiliki *file* .md seperti README.md yang berfungsi sebagai dokumentasi proyek. Daftar *file* atau folder yang akan diabaikan oleh sistem kontrol versi Git ditentukan dalam *file* .gitignore. Terakhir, .env.example merupakan contoh *file* untuk variabel lingkungan (*environment variables*).

Berdasarkan nama-nama *file* seperti vulnerable.html, vulnerable.py, sql-injection-examples.py, dan nama direktori DevSecOps-Demo-TA, proyek ini kemungkinan besar adalah sebuah aplikasi web yang dirancang khusus untuk tujuan edukasi dan demonstrasi. Secara lebih spesifik, fungsi proyek ini adalah untuk mendemonstrasikan kerentanan keamanan (*vulnerabilities*), di mana aplikasi ini sengaja dibuat memiliki celah keamanan (seperti SQL Injection) sebagai contoh

PERPUSTAKAAN UNIVERSITAS JENDERAL ACHMAD YANI YOGYAKARTA

nyata. Kedua, proyek ini digunakan sebagai studi kasus untuk mengimplementasikan praktik DevSecOps. Kehadiran Jenkinsfile dan Dockerfile menunjukkan adanya alur otomatisasi untuk *build*, *test* (kemungkinan termasuk *security scanning*), dan *deployment* aplikasi. Ketiga, proyek ini juga memiliki halaman web standar seperti index.html yang berfungsi sebagai antarmuka pengguna sederhana.

Secara keseluruhan, proyek ini terdiri dari 75 *file*. Rincian berdasarkan jenis (ekstensi) *file* adalah sebagai berikut: terdapat 16 *file* Sass (.sass), 10 *file* gambar (.jpg, .png), 16 *file font* (.eot, .svg, .ttf, .wog), 8 file JavaScript (.js), 5 *file* HTML (.html), 2 *file* SCSS (.scss), dan 2 *file* Python (.py). Selain itu, terdapat 9 *file* konfigurasi (.conf, .json, .dockerignore, .gitignore, Dockerfile, Jenkinsfile, LICENSE, .env.example), serta 7 *file* lain-lain (.css, .md, .sql, .txt).

3.3 ALAT PENELITIAN

Dalam penelitian ini, berbagai alat digunakan untuk mendukung proses pengujian dan pengembangan sistem. Alat utama yang digunakan meliputi sebuah laptop dengan spesifikasi tertentu serta tiga *Virtual Private Server* (VPS**)** yang digunakan sebagai lingkungan uji coba dan *deployment.*

3.4 PERANGKAT KERAS

Laptop digunakan sebagai perangkat utama untuk melakukan pengembangan, analisis kode, dan manajemen infrastruktur penelitian. Adapun spesifikasinya adalah menggunakan Prosesor Intel Core i5, dengan RAM 8 GB, *Storage* sebesar 128 GB SSD dan *eskternal* SSD sebesar 256 GB, menggunakan sistem operasi macOS. Laptop ini digunakan untuk menulis kode seperti pada **Tabel 3.1**, serta mengontrol dan mengelola VPS yang digunakan dalam penelitian ini. VPS yang digunakan dalam penelitian ini sebayak tiga unit dengan spesifikasi pada

PERPUSTAKAAN UNIVERSITAS JENDERAL ACHMAD YANI YOGYAKARTA

Tabel 3.1 Sepesifikasi VPS Penelitian

No	Prosesor	Ram	Penyimpanan	Sistem Pengoprasian	Perangkat Lunak
1	2 vCPU	4 GB	8 GB SSD	Ubuntu 22.04 LTS	Jenkins
2	2 vCPU	4 GB	8 GB SSD	Ubuntu 22.04 LTS	Sonarqube
3	2 vCPU	4 GB	8 GB SSD	Ubuntu 22.04 LTS	Docker

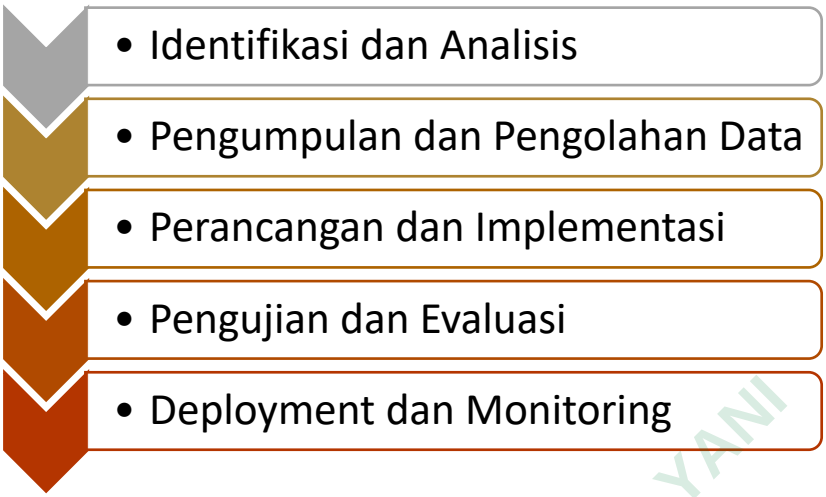
3.5 PERANGKAT LUNAK

Terdapat beberapa *software*, dalam penelitian ini di antaranya, VSCode sebagai penulisan *source code*, github sebagai penyimpanan *source code* dan pengintegrasian ke jenkins, SonarQube untuk menganalisis *source code* dan mengidentifikasi potensi kerentanan dalam aplikasi, Jenkins sebagai CI/CD untuk mengelola *pipeline,* docker untuk proses *build* dan *deployment* serta pengetesan aplikasi Web dan *Reverse Proxy.*

3.6 JALAN PENELITIAN

Jalan penelitian ini mencakup langkah-langkah yang dilakukan dalam pelaksanaan penelitian dengan pendekatan *Secure Software Development Life Cycle* (SSDLC) dan penerapan *Static Application Security Testing* (SAST) untuk memastikan keamanan *software* sejak tahap awal pengembangan. Setiap tahapan dalam penelitian ini disusun berdasarkan siklus pengembangan *software* yang terstruktur, seperti pada **Gambar 3.2**.

PERPUSTAKAAN UNIVERSITAS JENDERAL ACHMAD YANI YOGYAKARTA



Gambar 3.2 Jalan Penelitian

3.6.1 Tahap Identifikasi dan Analisis

Pada tahapan ini, dilakukan proses identifikasi permasalahan serta analisis kebutuhan sebagai fondasi dalam merancang dan mengembangkan sistem yang memiliki ketahanan terhadap berbagai ancaman keamanan. Kegiatan yang dilakukan mencakup tiga aspek utama. Analisis kebutuhan keamanan perangkat lunak dilakukan untuk memahami aspek-aspek kritis yang relevan dengan konteks penelitian. Dilakukan identifikasi terhadap sumber daya yang tersedia, termasuk perangkat keras dan perangkat lunak pendukung, seperti laptop pengembang dan server VPS yang digunakan sebagai lingkungan deployment. Lalu dipilih metodologi pengembangan yang sejalan dengan prinsip-prinsip *Secure Software Development Life Cycle* (SSDLC), serta dirumuskan strategi implementasi *Static Application Security Testing* (SAST) dengan memanfaatkan SonarQube, yang terintegrasi dalam *pipeline* Jenkins CI/CD guna memastikan otomasi dan konsistensi dalam proses pemeriksaan keamanan.

3.6.2 Tahap pengumpulan dan Pengolahan data

Tahapan ini difokuskan pada aktivitas pengumpulan data yang esensial dalam proses pengembangan serta pengujian sistem yang dirancang. Proses *akuisisi template* kode sumber dari repositori GitHub yang akan dijadikan landasan dalam

PERPUSTAKAAN UNIVERSITAS JENDERAL ACHMAD YANI YOGYAKARTA

proses pengembangan sistem, dilakukan pelaksanaan analisis terhadap arsitektur serta desain sistem guna memastikan bahwa sistem dirancang dengan mempertimbangkan aspek keamanan secara menyeluruh dan berlapis, lalu dilakukan pendalaman terhadap standar-standar keamanan aplikasi, termasuk pemahaman dan penerapan praktik *secure coding* sebagai bagian integral dalam pengembangan *software* yang aman.

3.6.3 Tahap Perancangan dan Implementasi

Tahapan ini merupakan fase inti dalam proses pengembangan sistem, yang dilaksanakan berdasarkan pendekatan *Secure Software Development Life Cycle* (SSDLC) serta integrasi metode *Static Application Security Testing* (SAST). Kegiatan utama dalam tahapan ini mencakup perancangan arsitektur sistem yang dirancang secara cermat dengan mempertimbangkan berbagai aspek keamanan guna menjamin perlindungan terhadap potensi kerentanan. Selanjutnya dilakukan proses pengembangan dan pengujian modul aplikasi dengan menerapkan teknik *secure coding*, sebagai upaya preventif terhadap eksploitasi kode oleh pihak yang tidak bertanggung jawab. Di samping itu, alat analisis keamanan SonarQube diintegrasikan ke dalam *pipeline* Jenkins, sehingga setiap perubahan pada kode program akan secara otomatis dianalisis untuk mendeteksi potensi celah keamanan sebelum aplikasi dipublikasikan ke lingkungan produksi (*deployment*).

3.6.4 Tahap Pengujian dan Evaluasi

Tahapan ini mencakup pelaksanaan pengujian menyeluruh terhadap aplikasi guna memastikan tingkat keamanan dan keandalan sistem sebelum diterapkan dalam lingkungan produksi. Prosedur yang dilaksanakan mencakup pengujian fungsionalitas aplikasi untuk memverifikasi bahwa seluruh fitur yang dikembangkan beroperasi sesuai dengan yang diharapkan. Selain itu, dilakukan pengujian keamanan menggunakan pendekatan *Static Application Security Testing* (SAST), di mana SonarQube dimanfaatkan untuk menganalisis kode secara statis dalam rangka mengidentifikasi potensi kerentanan. Hasil dari seluruh rangkaian pengujian kemudian dianalisis secara sistematis, dan apabila ditemukan kelemahan,

PERPUSTAKAAN UNIVERSITAS JENDERAL ACHMAD YANI YOGYAKARTA

dilakukan tindakan korektif guna memastikan bahwa sistem telah berada dalam kondisi optimal sebelum tahap *deployment*.

3.6.5 Tahap Deployment dan Monitoring

Setelah aplikasi berhasil melewati seluruh tahapan pengujian, proses *deployment* ke lingkungan produksi dilakukan dengan menerapkan mekanisme yang aman dan terstandarisasi. Tahapan ini mencakup penerapan aplikasi menggunakan container Docker yang dikonfigurasi dengan server Nginx sebagai *reverse proxy* guna mendukung pengelolaan lalu lintas jaringan secara efisien. Sistem kemudian didistribusikan kepada pengguna akhir setelah melalui proses verifikasi untuk memastikan bahwa seluruh komponen telah memenuhi standar keamanan yang ditetapkan. Selanjutnya, dilakukan aktivitas *monitoring* dan pemeliharaan secara berkala, termasuk penerapan pembaruan sistem yang bersifat preventif maupun responsif terhadap potensi kerentanan yang mungkin muncul difase akhir.

PERPUSTAKAAN UNIVERSITAS JENDERAL ACHMAD YANI YOGYAKARTA