Arkitektplan

Edvin Gibro
Bacilika Glansholm
Viktor Holta
Simon Karlsson
Jessica Kjellin
Max Randow
Erik Simonson
Jakob Söderström

8 mars 2024

Version 0.1

Status

Granskad	NAMN	2024-xx-xx
Godkänd	NAMN	2024-xx-xx

Projektidentitet

Kund: Ola Angelsmark, Advenica AB
E-post: ola.angelsmark@advenica.com

Handledare: Eric Ekström
E-post: eric.ekstrom@liu.se

Kursansvarig: Kristian Sandahl
E-post: kristian.sandahl@liu.se

Projektdeltagare

Namn	**Ansvar**	**E-post**
Edvin Gibro	Teamledare	edvgi966@student.liu.se
Bacilika Glansholm	Arkitekt, Vice teamledare	bacgl188@student.liu.se
Viktor Holta	Testledare	vikho305@student.liu.se
Simon Karlsson	Analysansvarig	simka157@student.liu.se
Jessica Kjellin	Kvalitetssamordnare	jeskj559@student.liu.se
Max Randow	Dokumentansvarig	maxra518@student.liu.se
Erik Simonson	Konfigurationsansvarig	erisi409@student.liu.se
Jakob Söderström	Utvecklingsledare	jakso277@student.liu.se

INNEHÅLL

DOKUMENTHISTORIK

Version	Datum	Utförda ändringar	Utförda av	Granskad
0.1	2024-02-19	Första utkast	PUM14	

1 SYFTE

Syftet med dokumentet är att sammanställa de arkitektoniska elementen hos projektet som utför automatisk säkerhetsanalys av mjukvara. Inkluderat i dokumentet är Mål, Filosofi, Antaganden, Beroenden samt mekanismer och översikter.

2 MÅL OCH FILOSOFI

2.1 Filosofin hos arkitekturen

Det som ska uppnås med arkitekturen är ett funktionellt och pålitligt system som ska vara enkelt att underhålla. Det ska även vara möjligt att vidareutveckla vid behov.

2.2 Mål hos arkitekturen

- Integrerbar: Arkitekturen ska kunna integreras som ett steg i en Git-pipeline.
- Underhåll: Arkitekturen ska kunna underhållas på lång sikt.
- Öppen licens: Projektet ska vara öppet och licenserat med GPLv3.

3 ANTAGANDEN OCH BEROENDEN

- Antagande: Användaren innehar grundläggande teknisk kompetens.
- Antagande: Server ska finnas tillgängligt för databasen.
- Beroende: Applikationen behöver internetuppkoppling för att hämta information om bibliotek.
- Beroende: Indatan måste vara en SBOM med CycloneDX-format som genereras externt.
- Beroende: Analysen av SBOMen kommer utgå ifrån OpenSSF Scorecards.

4 ARKITEKTONISKT SIGNIFIKANTA KRAV

De arkitektoniska signifikanta kraven kan ses i [1].

5 BESLUT, BEGRÄNSNINGAR OCH MOTIVERINGAR

- Beslut: Exekvera projektet i en Docker Container.
  - Motivering: Försäkra att resultaten inte påverkas av yttre faktorer.

- Beslut: Ha ett API lager mellan klient och server
  - Motivering: Underlättar vidareutveckling och omstrukturering.
- Beslut: Använda en *Microservice* arkitektur
  - Motivering: Ger snabb uppstart och simpliciteten underlättar testning samt samarbeten mellan olika utvecklingsgrupper.

6 ARKITEKTONISKA MEKANISMER

6.1 Arkitektonisk Mekanism 1: Microservice Arkitektur

- Syfte: Microservice arkitektur tillåter att olika moduler utvecklas individuellt. Dess simplicitet underlättar utveckling och testning då man kan testa modulerna separat från varandra
- Attribut: CI/CD, skalbarhet, uppbrytning av tjänster
- Funktion: Varje modul av applikationen ansvarar för en del av tjänsten, exempelvis resultatvisning och inhämtning av data.

6.2 Arkitektonisk Mekanism 2: Containerisering

- Syfte: Containerisering gör det möjligt att isolera och hantera applikationsomgivningar som i sin tur ger mer flexibilitet vid distribuering.
- Attribut: Isolering, konsekvent, portabilitet.
- Funktion: kapsulerar miljön, bibliotek och beroenden i en container inför exekvering.

6.3 Nyckelabstraktioner

- Användare: den som matar in en SBOM samt kravprofil.
- OpenSSF Scorecards: analyserar SBOMen och ger ett resultat
- Kravspecifierare: sållar resultatet baserat på kravprofil.

6.4 Lager hos det arkitektoniska gränssnittet

Applikationen följer ett microservice arkitekturmönster där resultatet av SBOMen kommer att ske lokalt med hjälp av OpenSSF Scorecards. OpenSSF Scorecards kommer i sin tur att hämta information om bibliotek från internet. Organisering och analys av resultatet kommer sedan att ske i en enda kodbas som är uppdelad i olika moduler baserat på vilket område de täcker. Resultatet kommer sedan visas för användaren genom ett gränssnitt där man kan interagera med gränssnittet för att få en mer detaljerad vy över specifika delar av resultatet.

7 ARKITEKTONISK ÖVERSIKT

7.1 Logisk översikt

Den logiska översikten av applikationen inkluderar en GUI där användaren matar in en SBOM samt bestämmer en kravprofil som SBOMen ska analyseras utifrån kravprofilen. Vidare interagerar GUI:n med en backend som analyserar datan och visar upp resultatet för användaren. Se [bilaga 1].

7.2 Operativ översikt

Den operativa översikten redogör för hur användarens indata matas in genom ett användargränssnitt och sedan hur applikationen hanterar indatan i de olika modulerna. Till sist skickas resultatet tillbaka till användargränssnittet för att visas upp för användaren. Se [bilaga 2].

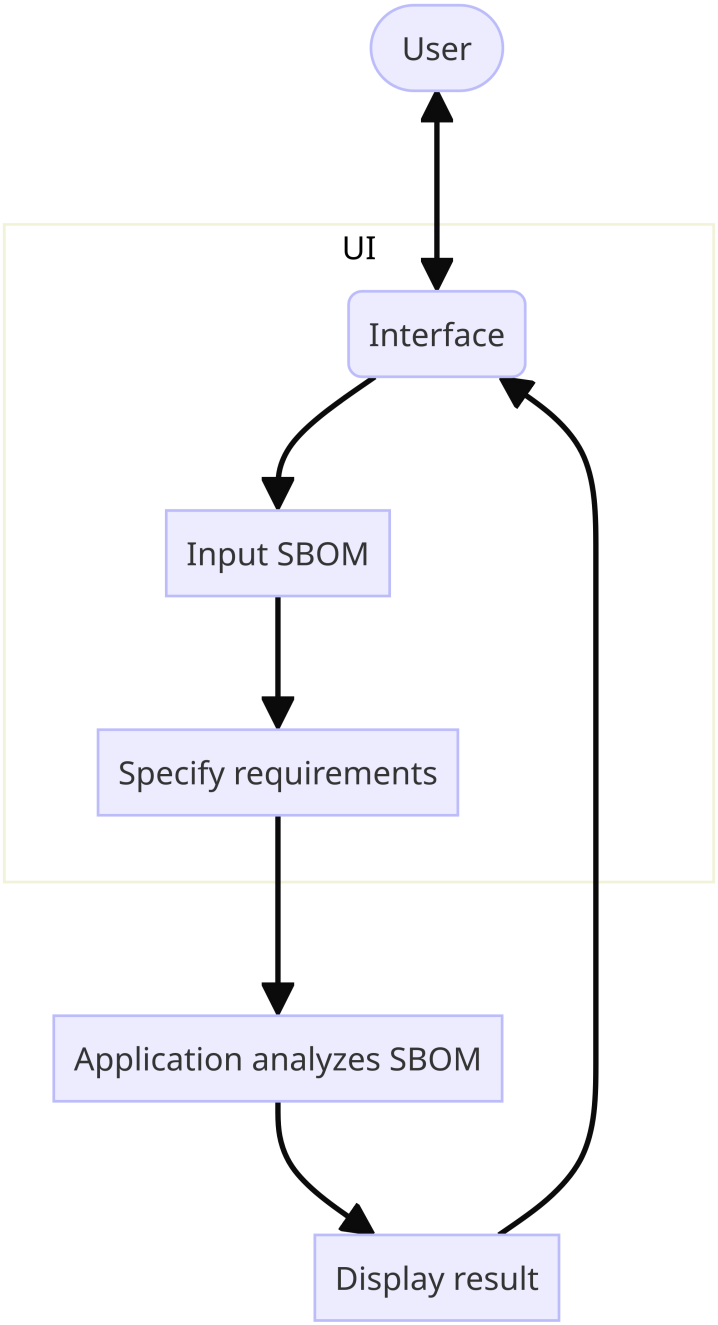
7.3 Användarfall

Användarfallen listar olika användarscenarion som nedladdning och uppstart, inladdning av SBOM, kravspecifiering och resultatuppvisning. Se [Bilaga 3]

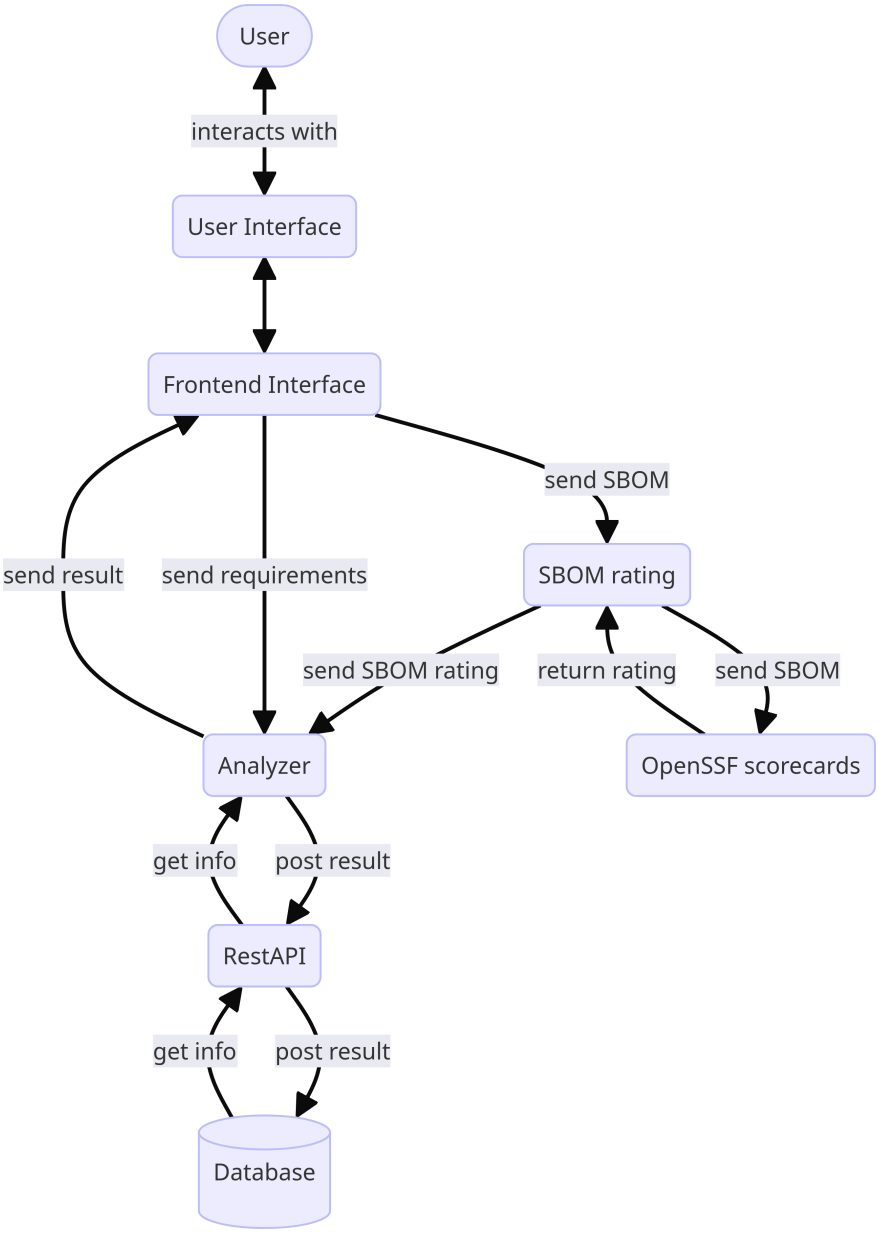
8 BILAGOR

REFERENSER

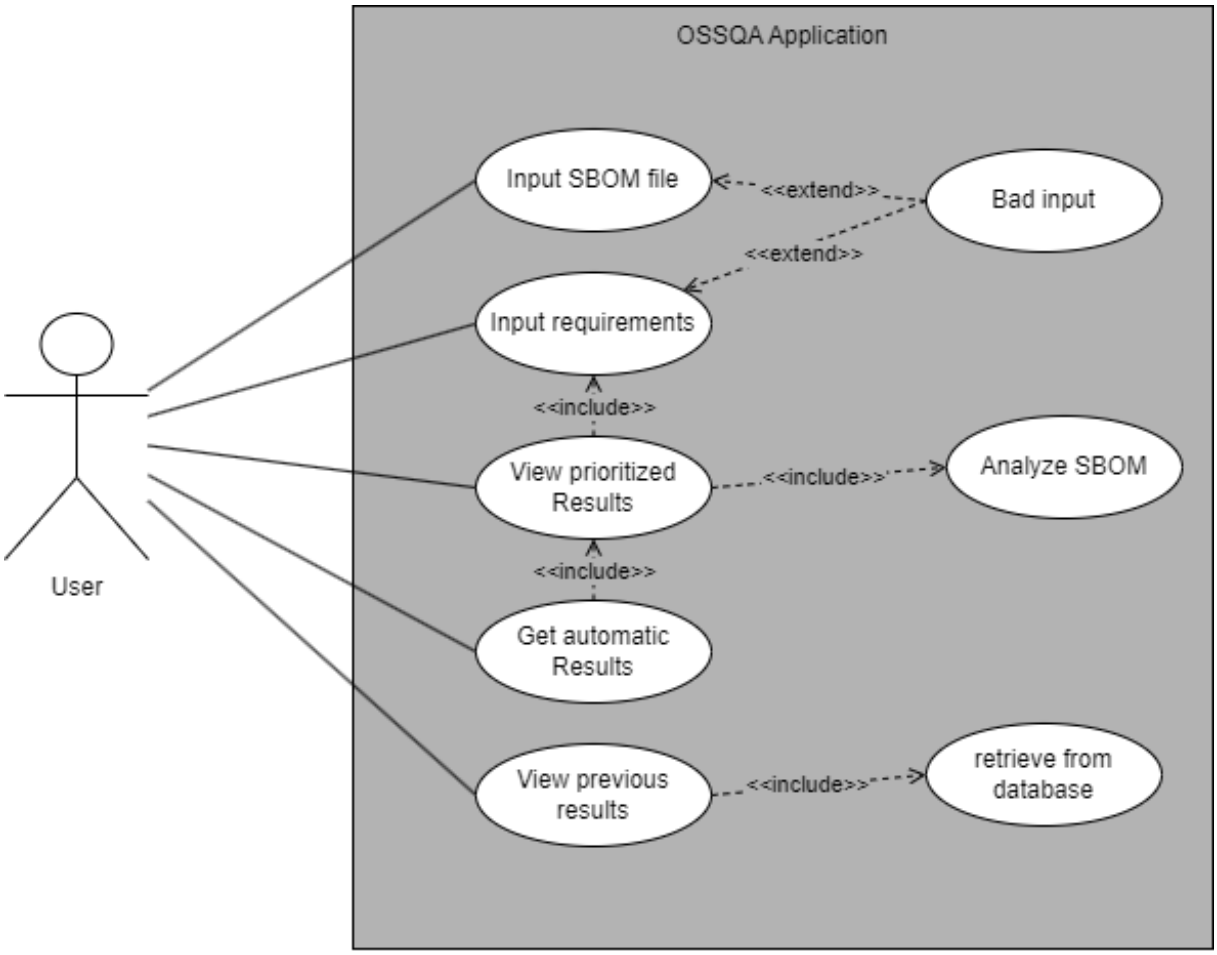
[1] PUM14, “Kravspecifikation,” https://github.com/OSSQA-PUM/OSSQA/blob/main/documents/TDDD96_Kravspecifikation.pdf, 2024.



Figur 1: Den logiska översikten av arkitekturen



Figur 2: Den operativa översikten av arkitekturen



Figur 3: Användarfallen hos applikationen