

Harnessing GenAI & Open Frameworks:

Transforming Engineering Teams
into Strategic Innovators
and Business Value Creators

Speaker:

Chandrakanth Paladugu {Chandu}

Director of Cloud Native, DevSecOps & Quality Engineering



Agenda



**Education on
GenAI**

**GenAI for
Modernization**

Use Cases

Possibilities



**Sharing today's
experience –
looking at it as an
Eng.**

Setting Context

Services – Application Development & Managed Services



AI Product - Defects & Anomaly detection for Business



Product & SaaP (Service as a Platform)
- New model for Services + CodeGen & Platform Engineering product - - SaaS 2.0



GenAI is a Tech



- Foundational Models with complex architectures & algorithms
 - Compute Power
 - Data
-
- GenAI tech depends on the type of Data representation & relationships
 - GenAI is trained on large datasets as general-purpose model using Transformers Modeling for self-supervised capabilities
 - GenAI needs to be treated as tech, not as a direct solution

Unlocking GenAI's Potential Responsibly

- Considerations



Image by Freepik

- Prioritizing Business Requirements
- Skill debt
- GenAI System Development Life Cycle (SDLC)
- GRC (Governance, Risk & Compliance), Transparency, Trust, Software Supply Chain Security
- Data Protection & Responsible AI Strategies
- Safe & Secure In-House AI Platforms
- Evolving from AI Users to AI Value Creators

GenAI Usage – Business perspective



Image by Freepik

- **Solve Real Business Problems like App Modernization, Enhancing Customer Services or new Offerings**
- **Create new vital Solutions to improve business value**
- **Encourage Innovation by leveraging AI**
- **Create new line of business for new revenue generation**
- **Streamlining Operations & Processes**

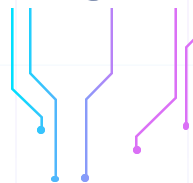
GenAI Usage - Eng. Perspective



Image by Freepik

- **App & Infra Modernization**
- **New Tech. adoption**
- **Rebuilding existing applications with modern secure languages like Go & Rust**
- **Spending more time on understanding and adding value to the business**
- **Being Agnostic & Future Ready**

Navigating Complexities - GenAI Adoption Challenges



**Code
Ownership**

**Speed of adoption vs Treating
AI as journey for Long-Term
Benefits**

**GenAI Quality
Engineering**

**Standards: Software Supply
Chain Security, DevSecOps, etc.**

FinOps

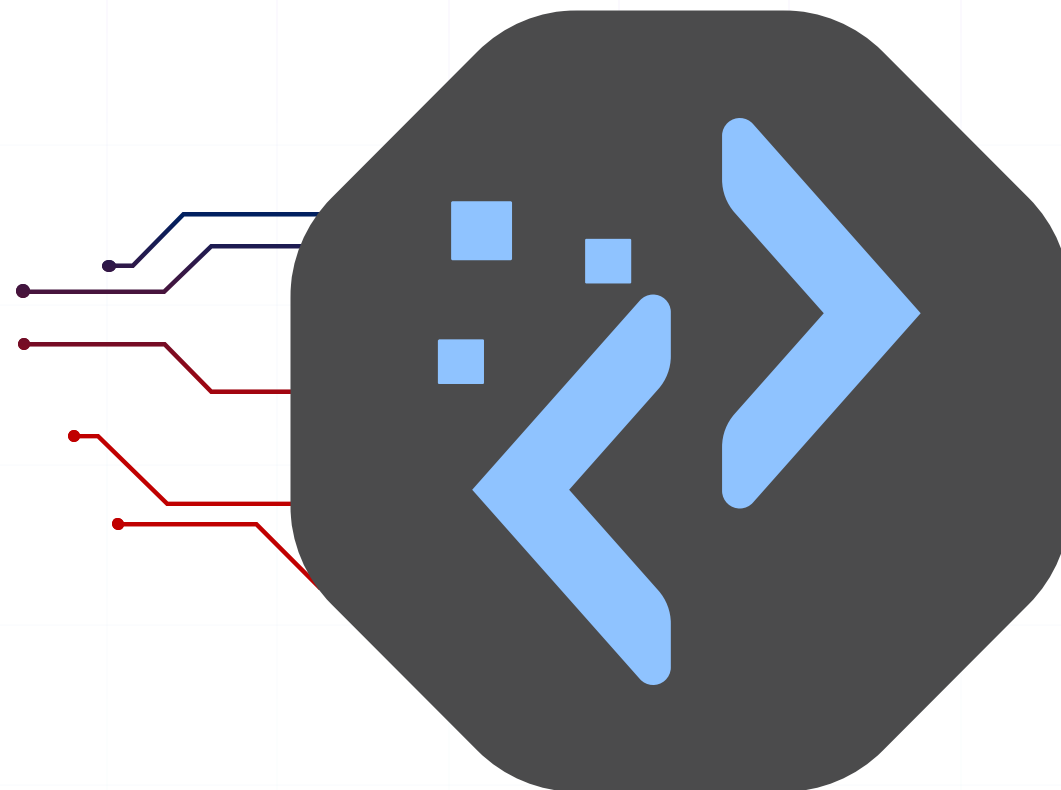
**External
data Risks**

AI "Hallucination"

**Lack of Global Industry Standards:
Trust, Transparency, and Bias Reduction**

Introducing Agnostic Open Framework "Compag" - Enabling Engineering

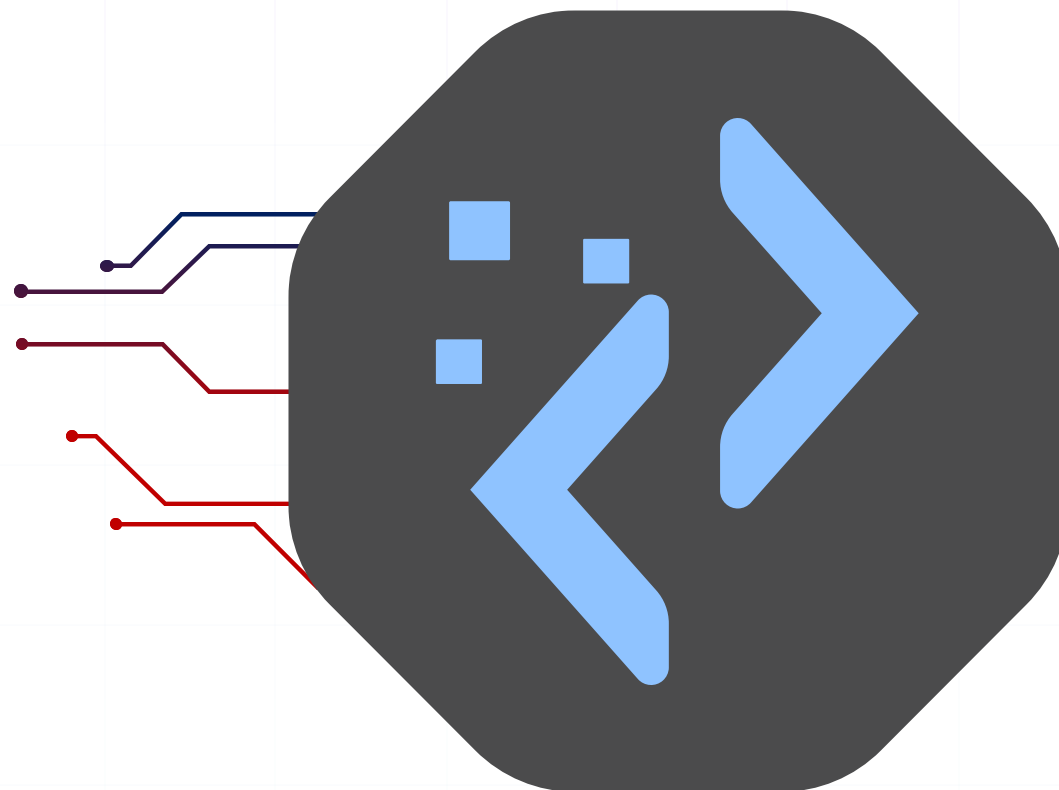
- Own Your Code with Compag License
(Privatization of Code, Copyright)
- Standardize & Automate Code Generation
- Reduce Development Time by 50%
- Generate Code in Multiple Programming Languages
– Golang, C#, Java, Rust, Python, etc.
- Pair it With GenAI for Business Use Cases



Introducing Agnostic Open Framework "Compage" - Enabling Engineering

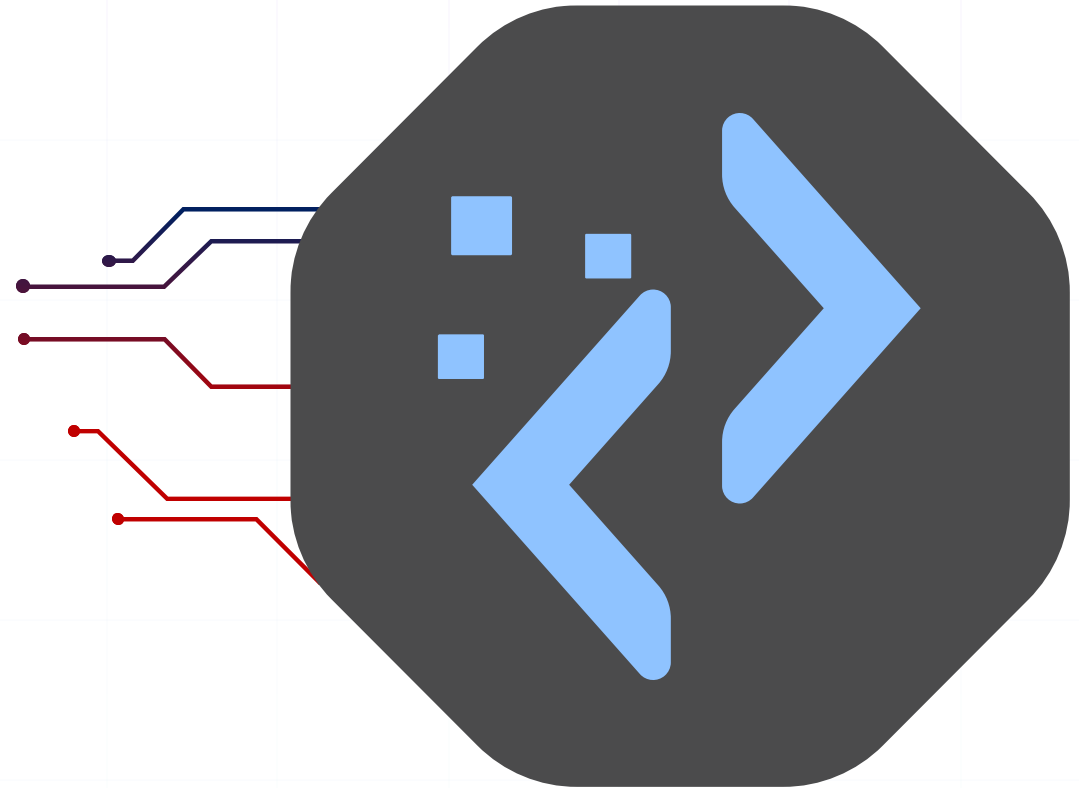
More Benefits:

- Auto Self-Healing, Updating & Self-Maintenance
- Auto Enhancements of Security Threats & Proactive Preparation of Evolving Future Threats
- Focus on Reducing Skill Gaps
- Drive Business Innovation and Iteration
- Training Engineers as Business Innovators



Here's a quick demo of how
Compage saves development time
while following standards

Quick flow of it



Input the Reqs

Node Properties : node-66

Name of Component *
Security

Language *
dotnet

☒ REST Server

Template *
compage

Framework *
dotnet-clean-architecture

Port *
8080



☒ SQL DB

SQL Database *
MSSQL

☐ NoSQL DB

ADD RESOURCE

Existing REST resources :

Resource	Actions
AuthenticateUser	 

☐ gRPC Server

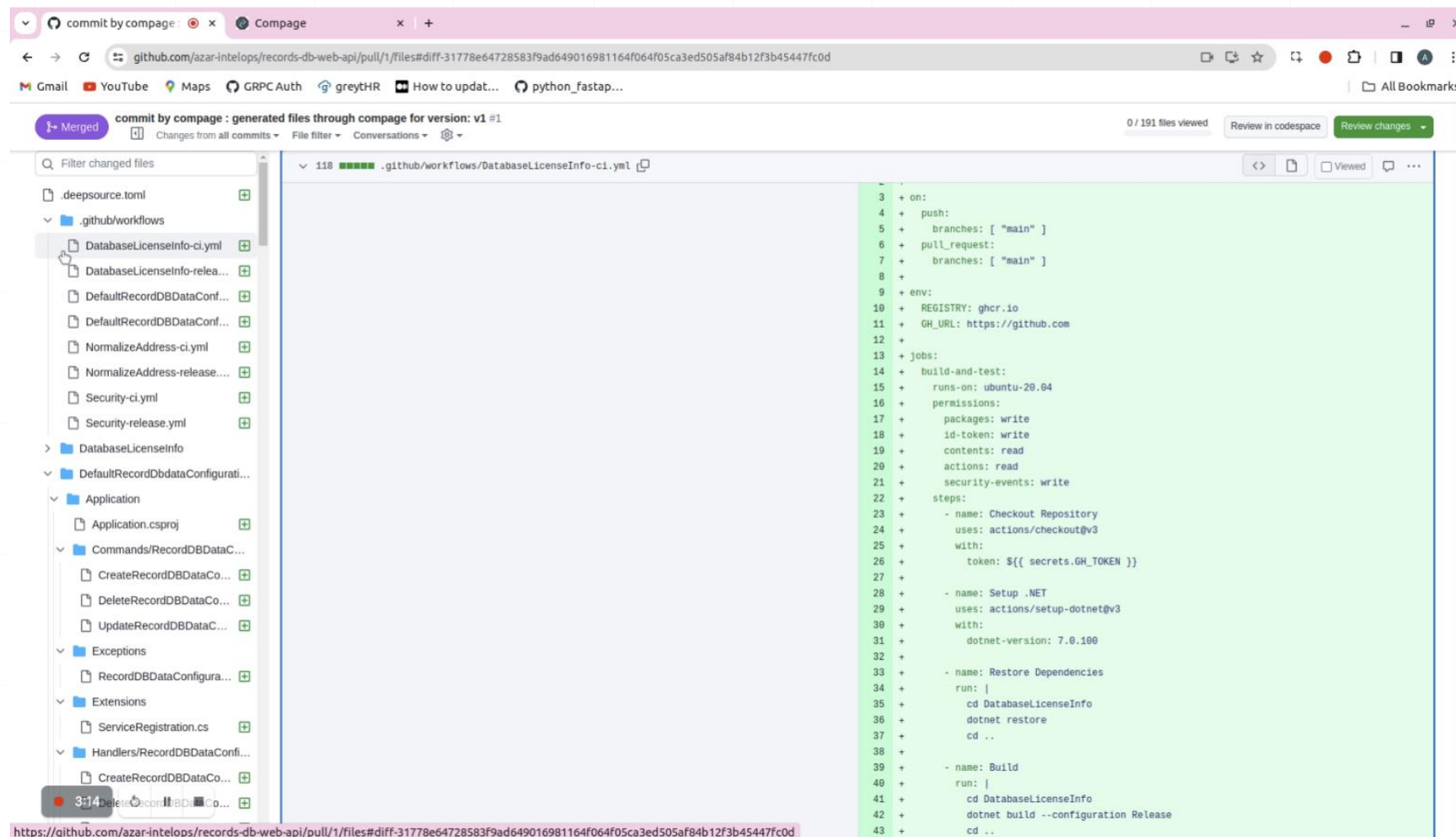
CANCEL UPDATE NODE

- Initialize a project, specifying its service name and associating it with a Git repository.
- Utilize Compage diagramming canvas to represent each microservice as a distinct node.
- Define specific requirements (such as programming language, database, and framework) within node properties to facilitate code generation.



Generated Code

Along with code-base, Compage automatically creates CI/CD pipelines, containerizes applications, infrastructure manifests and offers a flexible, agnostic setup.



The screenshot shows a GitHub pull request interface. The browser address bar displays the URL: `github.com/azar-intelops/records-db-web-api/pull/1/files#diff-31778e64728583f9ad649016981164f064f05ca3ed505af84b12f3b45447fc0d`. The page title is "commit by compage : generated files through compage for version: v1 #1". The left sidebar shows a file explorer with the following structure:

- `.deepsource.toml`
- `.github/workflows`
 - `DatabaseLicenseInfo-ci.yml` (selected)
 - `DatabaseLicenseInfo-relea...`
 - `DefaultRecordDBDataConf...`
 - `DefaultRecordDBDataConf...`
 - `NormalizeAddress-ci.yml`
 - `NormalizeAddress-release...`
 - `Security-ci.yml`
 - `Security-release.yml`
- `DatabaseLicenseInfo`
- `DefaultRecordDbdataConfigurati...`
- `Application`
 - `Application.csproj`
 - `Commands/RecordDBDataC...`
 - `CreateRecordDBDataCo...`
 - `DeleteRecordDBDataCo...`
 - `UpdateRecordDBDataC...`
 - `Exceptions`
 - `RecordDBDataConfigura...`
 - `Extensions`
 - `ServiceRegistration.cs`
 - `Handlers/RecordDBDataConf...`
 - `CreateRecordDBDataCo...`
 - `DeleteRecordDBDataCo...`
 - `UpdateRecordDBDataC...`

The main content area shows a diff for the file `.github/workflows/DatabaseLicenseInfo-ci.yml`. The diff shows the following content:

```
3 + on:
4 +   push:
5 +     branches: [ "main" ]
6 +   pull_request:
7 +     branches: [ "main" ]
8 +
9 + env:
10 +   REGISTRY: ghcr.io
11 +   GH_URL: https://github.com
12 +
13 + jobs:
14 +   build-and-test:
15 +     runs-on: ubuntu-20.04
16 +     permissions:
17 +       packages: write
18 +       id-token: write
19 +       contents: read
20 +       actions: read
21 +       security-events: write
22 +     steps:
23 +       - name: Checkout Repository
24 +         uses: actions/checkout@v3
25 +         with:
26 +           token: ${ secrets.GH_TOKEN }
27 +
28 +       - name: Setup .NET
29 +         uses: actions/setup-dotnet@v3
30 +         with:
31 +           dotnet-version: 7.0.100
32 +
33 +       - name: Restore Dependencies
34 +         run: |
35 +           cd DatabaseLicenseInfo
36 +           dotnet restore
37 +           cd ..
38 +
39 +       - name: Build
40 +         run: |
41 +           cd DatabaseLicenseInfo
42 +           dotnet build --configuration Release
43 +           cd ..
```

License Notice

Generated code is licensed as per your requirement, use Compage to add your copyright notice, incorporating custom language in the language model.

The screenshot displays a GitHub repository page for `azar-intelops/records-db-web-api`. The page shows a merge pull request #1 from `azar-intelops/main-v1`. The file list includes `.github/workflows`, `DatabaseLicenseInfo`, `DefaultRecordDbdataConfiguration`, `NormalizeAddress`, `Security`, `.deepsource.toml`, `LICENSE`, and `README.md`. The README is selected, showing the 'Code generated by compage' section and 'Steps required for running the GitHub actions for this project.' The license is Apache-2.0.

Code generated by compage

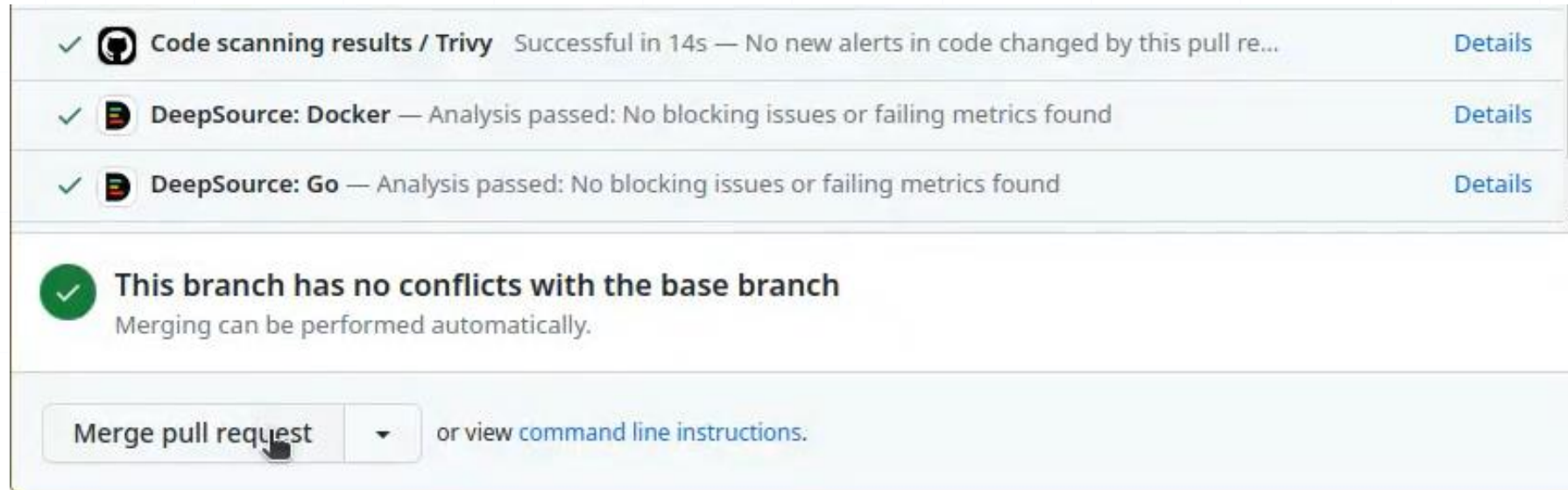
github.com/azar-intelops/records-db-web-api

Steps required for running the GitHub actions for this project.




- Create a GitHub token by following the steps given [here](#), you need to have these scopes for the token [repo, write:packages]
- Add the above token as an environment variable [GH_TOKEN] to this project, please follow the steps given [here](#)

Security Standards and Updates

Compage team continuously updates the code generation dataset to feed that dataset to generation-engine, which helps to keep the generated code up-to-date with standards and security. The generated codebase is automatically pushed to your specified repository, complete with security scanning, containerization, dependency setup, and passed tests.

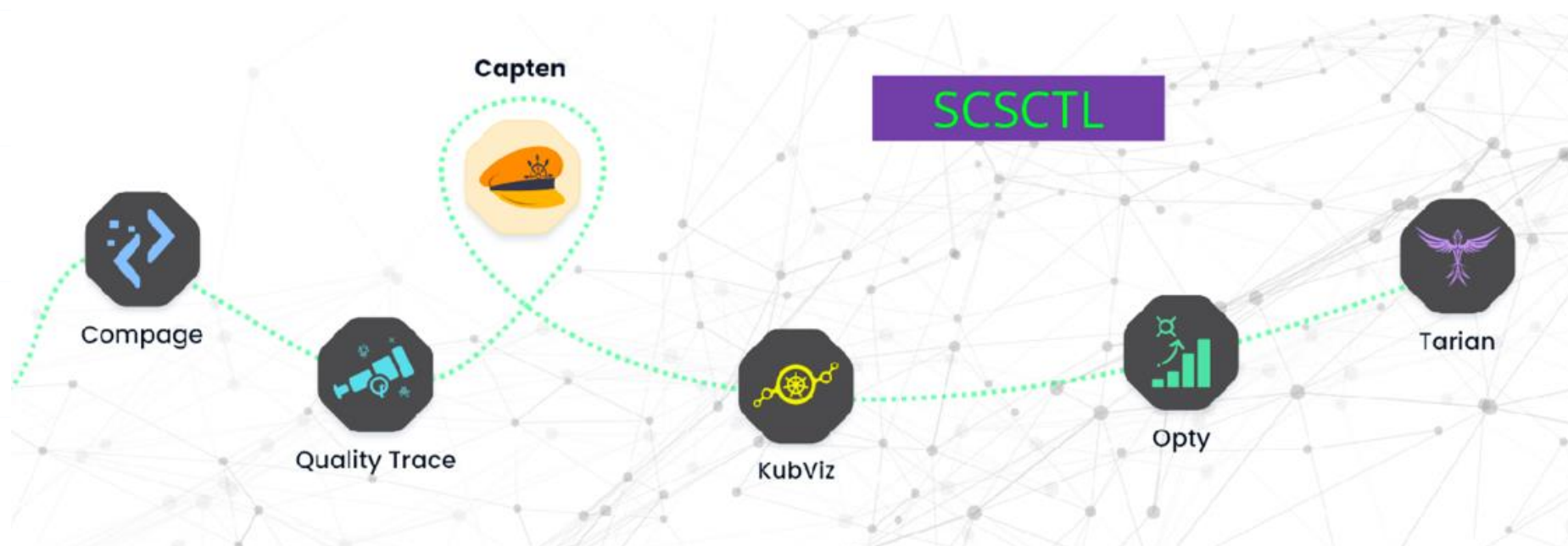


The screenshot displays the status bar of a GitHub pull request. It features three green checkmarks indicating successful security scans: Trivy, DeepSource: Docker, and DeepSource: Go. Below these, a green checkmark confirms that the branch has no conflicts with the base branch. At the bottom, a 'Merge pull request' button is visible, along with a link to view command line instructions.

✓  Code scanning results / Trivy Successful in 14s — No new alerts in code changed by this pull re...	Details
✓  DeepSource: Docker — Analysis passed: No blocking issues or failing metrics found	Details
✓  DeepSource: Go — Analysis passed: No blocking issues or failing metrics found	Details
✓ This branch has no conflicts with the base branch Merging can be performed automatically.	
Merge pull request or view command line instructions .	

What more can Compage do?

Capten Stack - Built to ALWAYS Stay with You!



Dev

Ops

Sec

Take Aways

- Impact:
 - Hands-on Style
 - Jobs & Pay
 - Deliverables Expectations
 - Skills Expectations
 - Management Changes
 - New Job Roles & Responsibilities
- Advice:
 - Learning Approach
 - Strong with Fundamentals
 - Learning curve
 - Up Skilling
 - Understanding Business
 - Adding Value to Business

Connecting dots to the Context that was set in the beginning

New Services Approach:

SaaP (Service as a Platform or Service as a Product)

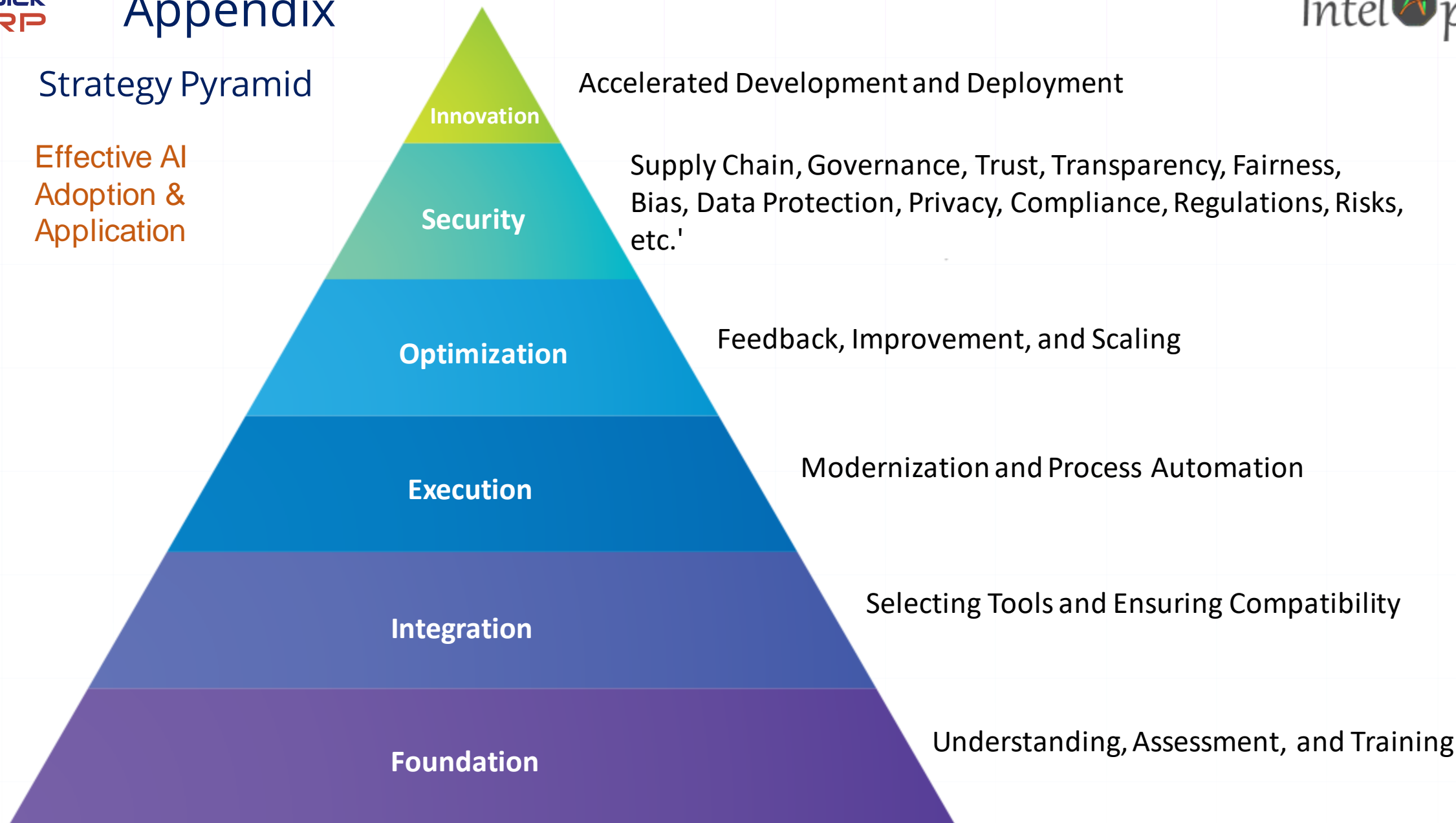
SaaP aliases:

- SaaS 2.0 (Service as a Software)
- SaaS or SvaaS (Service via a Software)



Strategy Pyramid

Effective AI
Adoption &
Application



Modernization Use Case



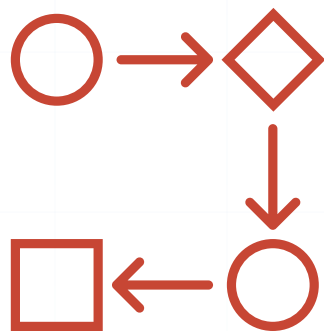
- **Dependencies:**

- Complexity, Heterogeneity, Relationships between Services & Projects

- **Modernization Benefits:**

- To Enhance Business Value
- For Improved Security and Cost Efficiency
- To Gain Competitive Edge
- To Foster Business Innovation

Modernization Use Case

• **Challenges:**

- Documentation and Knowledge Gaps in Legacy/Monolith Apps
- Skill Gaps and Securing Skilled Talent
- Cost-Effective Strategies for Modernization
- Adopting New Technologies Efficiently

• **Phases:**

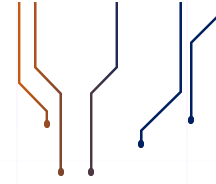
- **Planning Phase:** Strategy Development, Requirements Analysis, and Deciding What to Keep or Remove from the Existing Code Base
- **Implementation Phase:** Employing Standard Approaches such as DevSecOps, Software Supply Chain Security, Ensuring Observability, Governance, Risk and Compliance (GRC), Trust, Transparency, and Addressing Copyright Issues.

Modernization Use Case

App Modernization & Its Direct Relation to Infra
Modernization:

- Difficulty in Aligning with Industry Standards: NIST, CIS, FIPS, Open API, Open Banking, and evolving financial and supply chain security standards.
- Integration with other business partners and security and privacy challenges
- Skills gap in the Finance domain





Thank you!

Q & A

<https://github.com/intelops>

<https://github.com/intelops/compage>



AI

New Era in
Technological
Innovation



{Agnostic Framework}

New Era in
Technological
Implementation

