Architecture Vision Document for AndersenSite

Issued: 07/07/2021

Version: 1.1.0

Andersen 75 Zhylianska Street, Floor 3, Eurasia Business Center, Kyiv, Ukraine

Revision History

Date	Version	Description	Author
12.06.2021	0.1.0	Analyze and fill architectural drivers	Dmitry Stelmakh
13.06.2021	0.2.0	Make changes to architectural drivers	Dmitry Stelmakh
19.06.2021	0.3.0	Fill prioritized Quality attributes and Quality attribute scenarios	Dmitry Stelmakh
20.06.2021	0.4.0	Solution context view	Dmitry Stelmakh
26.06.2021	0.5.0	Container view	Dmitry Stelmakh
26.06.2021	0.6.0	Monolithic Backend and API component view	Dmitry Stelmakh
27.06.2021	0.7.0	Software deployment view	Dmitry Stelmakh
27.06.2021	0.8.0	Implementation roadmap	Dmitry Stelmakh
07.07.2021	1.1.0	Operation plan	Dmitry Stelmakh

Table of contents

Table of contents

Table of contents

Introduction

1.1 Vision Purpose

This document provides a high-level description of the technical solution for the AndersenSite project. This solution was conceived to address all requirements described by the AndersenSite Vision & Scope Document. The current document describes the major modules that will make up the solution within the initial scope as well as the interaction and integration aspects.

This document is intended to:

- overview the high-level project description;
- provide a visual representation of interconnections between the application core services;
- describe delivery and operations processes;
- provide some explanations for the decisions made;
- recommend a technology stack;
- support on-boarding processes for new members of the development team.

Any changes made to the document after agreeing on the initial version should be tracked within the Revision history table.

1.2 Business Context

Andersen is an international software development company with deep expertise in modern application development for a wide range of industries. Andersenlab is a SPA responsible for providing an overview of companies and their activities. It also encourages potential clients to order Andersen's services and creates a positive public image of the company.

The system allows a potential client to make a strong impression with a well- designed introduction for business.

After releasing the initial version of the system and analyzing questions asked by the clients, it has been decided to redesign the system. While using the system, clients were not always able to find the information they needed and had to often contact the helpdesk. As a result, some of them prefer to cooperate with the company's competitors offering fuller information about their services and experience via their websites.

It is necessary to enhance the conversion of existing referrals as most potential clients inevitably do their research by stacking Andersen against its competitors. That's why improvements on the site are aimed at enhancing the visual and essential components of the content and expanding the functionality and capabilities available to users to increase the conversion of users to leads and, accordingly, to potential clients of the company.

Executive Summary

2.1 High Level Description

To attract more potential clients, it is necessary to make the system intuitive and easy to use. The system should be redesigned in such a way that the potential client will want to start collaborating with the company.

In addition, it takes the sales department a lot of time to support the communication process with potential clients. Because of that, the sales center’s resources are spent less productively. Andersenlab should help to save that time by providing answers to common questions and inquiries. It is necessary to add functionality that potential clients are most often interested in.

2.2 Key Decision

There are key decisions that were done during the discovery phase:

- Base technology stack: JavaScript and PHP
- Backend: Laravel
- Frontend: Gatsby.js
- Deployment to AWS

2.3 Key Risk

These are key risks identified at the discovery phase:

- It must be defined how to forecast and measure the achievement of business objectives;
- Implementation efforts may vary due to the high-level elicitation of requirements.

3.1 System Overview

This section describes the general function and purpose of the system or subsystem the architecture of which is described in this Architecture Vision Document.

3.2 Business Goals

The section enumerates essential business goals for the solution.

ID	Business Goal Description
BG-1	Increase the number of leads to 300% within 1 year (from the site) after the deployment of the new version of the system.
BG-2	Initial phase should comprise all the features approved for Milestone 1
BG-3	Quality must meet the Customer’s requirements and success criteria.
BG-4	Planned team: Business analyst, UI/UX Designer, Frontend developers, Backend developers, DevOps engineers, QA engineers, Project manager
BG-5	Reduce the cost of the high-quality leads by 2,5 times within 1 year after the deployment of the new version of the system.
BG-6	By increasing the quality of incoming information from leads, reduce the time spent by a sales specialist on processing it by 25% within 1 year after the deployment of the new version of the system.

3.3 Major Features

This section enumerates the major features of the solution.

ID	Feature Description
MF-1	CV
MF-2	Cases
MF-3	Customers feedbacks
MF-4	Corporate blog
MF-5	Pricing
MF-6	Services
MF-7	Real time projects map

3.4 Design Constraints

This section lists the constraints exercising significant influence over the architecture projected for the designed solution. These can be of business, technical, resource, and other types.

ID	Description
CT-1	The system will be available only as a web application.
CT-2	The system will have the capability to send text messages only.
CT-3	A user interface will be implemented only in English and German.
CT-4	The system will not have an admin page - all content is added manually by the dev-team.
CT-5	All databases should be located on Amazon servers.
CT-6	The system should support Android platform 8.0 version and higher, iOS 13.0 version and higher, Windows 10 and higher, MacOS 12 version and higher.
CT-7	Back-End technology requirements: PHP (7.4), MySQL (5.5-8.0), Laravel (8.4)
CT-8	Front-End technology requirements: Javascript (ECMAScript 6-11), HTML5, CSS3, SASS, Blade, Gulp
CT-9	WebServer technology requirements: Nginx

Architectural Drivers

CT-10	Browser versions requirements: • Chrome - 84.0.4147 and the latest version • Safari - 10.3.3 and the latest version • Mozilla Firefox - 66.0 and the latest version • Internet Explorer - 11 and the latest version • Microsoft Edge - 44.17763.1.0 and the latest version • Opera - 52 and the latest version

CT-11	Screen resolutions requirements:
	• Desktop:
	  ◦ 1920 x 1080
	  ◦ 1366 x 768
	  ◦ 1536 x 864
	  ◦ 1440 x 900
	• Mobile:
	  ◦ 375 x 667
	• Tablet:
	  ◦ 960 x 577

3.5 Architecture-related Concerns

The section describes a list of generic and specific architecture concerns. It can include Architectural requirements, practices, questions, issues, and risks important for the system design.

ID	Description
CN-1	Logging and monitoring
CN-2	Keep customer data secured
CN-3	Existing company blog based on WordPress
CN-4	Define Framework/library usage for frontend
CN-5	Laravel usage for backend
CN-6	Build, release, run
CN-7	Dev/prod parity
CN-8	Initial application already exist, should be defined - is it possible to continue development base on it
CN-9	Unit testing

3.6 Quality Attributes and Quality Attribute Scenarios

A Quality Attribute Scenario is an unambiguous and testable requirement for one or more Solution Quality Attributes such as Performance, Usability, Maintainability and others. The scenario consists of six parts: Source of Stimulus, Stimulus, Environment, Artifact, Response, testable and accurate Response Measure.

This section lists and prioritizes the scenarios pertinent to the designed solution

ID	Quality attribute	Description	Business Priority	Complexity
QA-1	Security	Solution have to be hosted on the AWS Cloud using Cloud architecture best practices	High	High
QA-2	Operability (Usability)	System supports several languages (English and German)	Medium	Low
QA-3	Operability (Usability)	Web Application should support responsive design for mobile and tablet devices (CT-11)	Medium	Medium
QA-4	Operability (Usability)	Web Application should support browser versions listed in CT-10	Medium	Medium
QA-5	Testability (Maintainability)	Unit and integration tests successful completion and code coverage should be as high as at least 70%	Low	Low
QA-6	Deployability	Code and architecture changes tested and provisioned on all environments during 1 day	Low	Medium

Solution Architecture

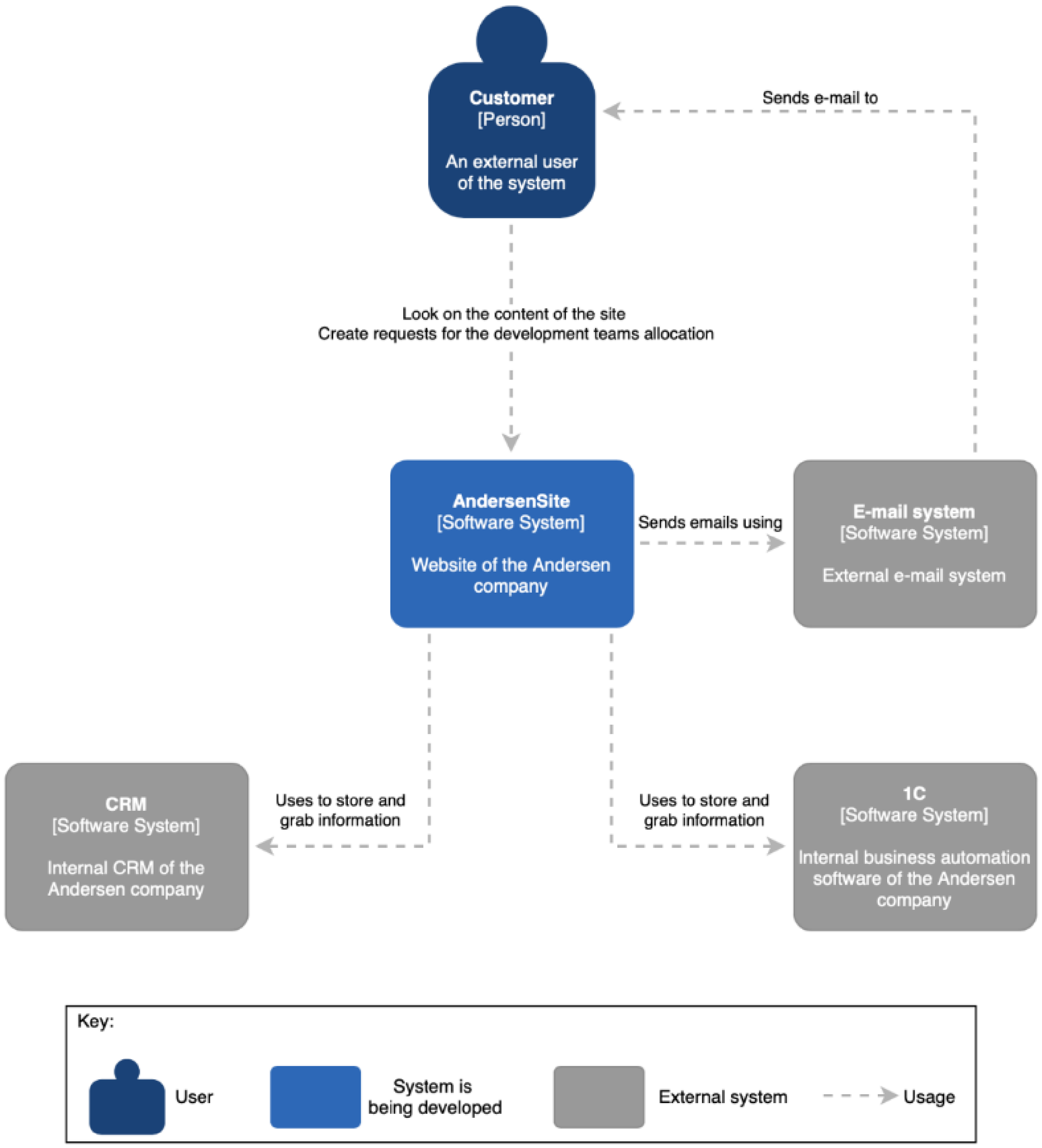
The Solution Architecture section is intended primarily for the Architecture Vision document. It defines and specifies the solution architecture design based on the architecturally significant requirements and constraints identified by the section Architectural Drivers.

4.1 Solution Context View

4.1.1 Intent

The view defines the primary solution components collaborating with the external systems and services. It is driven by the Business Case.

4.1.2 Representation



4.1.3 Elements

Table of annotated elements:

Name	Description
Customer	An external user of the system
AndersenSite	Website of the Andersen company
E-mail system	External e-mail system
CRM	Internal CRM of the Andersen company
1C	Internal business automation software of the Andersen company

4.1.4 Rationale

The section above provides a high-level view of what kind of system needs to be designed, and also gives an opportunity to view the scale of the application by displaying third-party / additional systems and services.

4.2 Container View

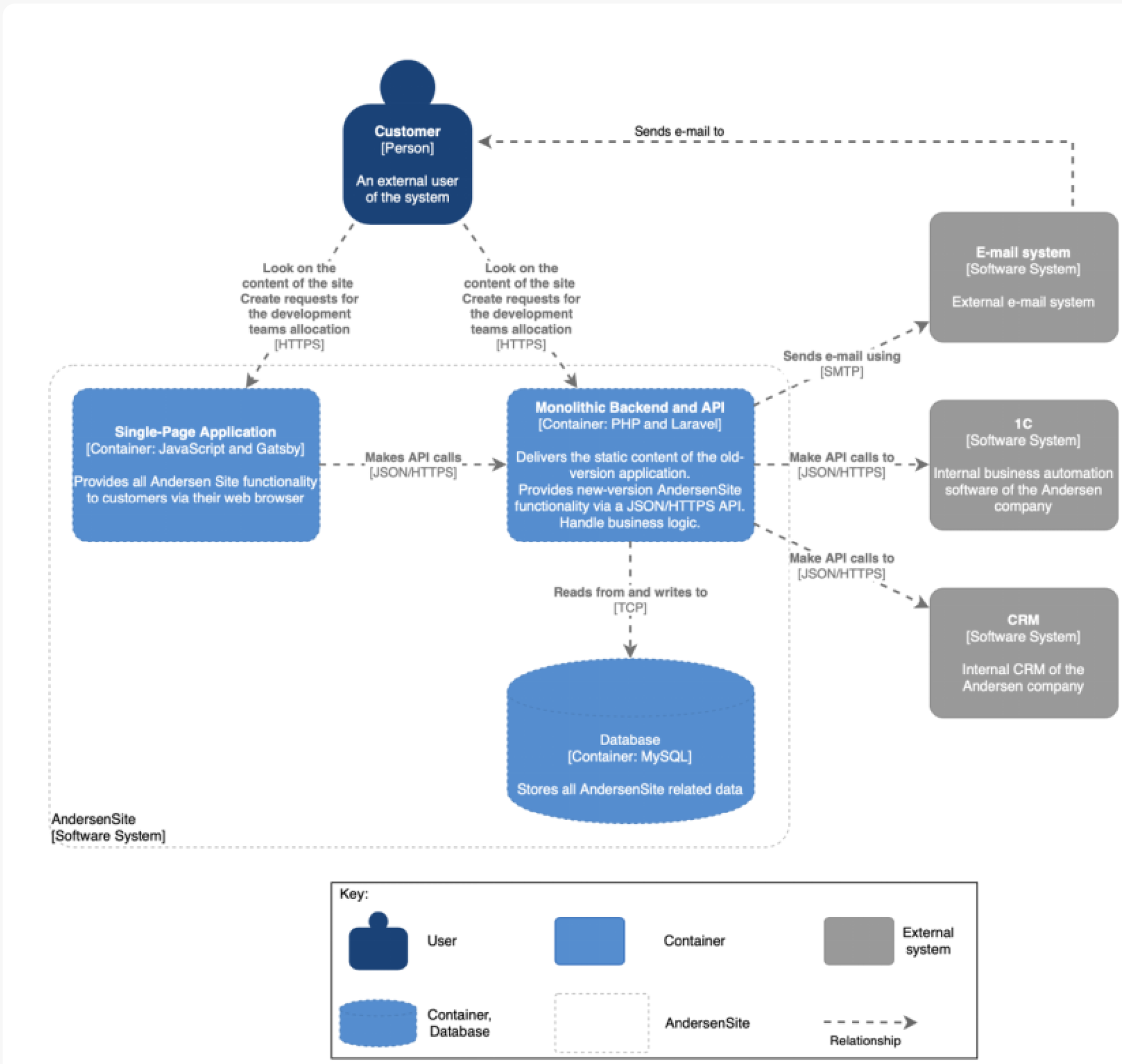
4.2.1 Intent

The view defines the runtime decomposition of all internal parts of the solution. It is driven by the SAR's and the architecture best practices applicable to cloud-based applications.

4.2.2 Context

The view context is defined by the Solution Context view where this section represents decomposition of the AndersenSite component.

4.2.3 Representation



4.2.4 Elements

Table of annotated elements:

Name	Description
Customer	An external user of the system
AndersenSite	Website of the Andersen company
Single-Page Application	Provides all Andersen Site functionality to customers via their web browser
Monolithic Backend and API	Delivers the static content of the old-version application. Provides new-version AndersenSite functionality via a JSON/HTTPS API. Handle business logic.
Database	Stores all AndersenSite related data
E-mail system	External e-mail system
CRM	Internal CRM of the Andersen company
1C	Internal business automation software of the Andersen company

4.2.5 Rationale

Client Server Architecture is a computing model via which the server hosts, delivers and manages most of the resources and services to be used by the client. This type of architecture has one or more client computers connected to a central server over a network or internet connection. This system shares computing resources. The client/server architecture is also known as a networking computing model or client/server network because all the requests and services are delivered over a network.

3-tier Client-Server architecture advantages:

- Improved Data Sharing: Data is retained by usual business processes and manipulated on a server and is available for designated users (clients) over an authorized access. The use of Structured Query Language (SQL) ensures open access from all client aspects as well as transparency in network services indicating that similar data is being shared among users.

- Integration of Services: Every client is given the capability to access corporate information via the desktop interface eliminating the necessity to log into a terminal mode or another processor.

- Shared Resources amongst Different Platforms: Applications used for client/server model are built regardless of the hardware platform or technical background of the entitled software (Operating System S/W). As a result, an open computing environment is provided for, enabling users to obtain the services from clients and servers (database, application, communication servers).

- Inter-Operation of Data: All development tools used for client/server applications access the back-end database server through SQL, an industry-standard data definition and access language, helpful for consistent management of corporate data. Advanced database products enable users/applications to gain a merged view of corporate data dispersed over several platforms. Rather than a single target platform, this ensures database integrity with the ability to perform updates on multiple locations enforcing quality recital and recovery.

- Data Processing capability regardless of the location: We live in an era which undergoes a transformation of machine-centered systems into user-centered systems. Machine-centered systems like mainframe, mini-micro applications had unique access platforms and functionality keys, navigation options, performance and security were all visible. Through client/server users can directly log into a system despite the location or technology of the processors.

- Easy maintenance: Since client/server architecture is a distributed model representing dispersed responsibilities among independent computers integrated across a network, it’s an advantage in terms of maintenance. It’s easy to replace, repair, upgrade and relocate a server while clients remain unaffected. This unawareness of change is called encapsulation.

- Security: Servers have better control access and resources to ensure that only authorized clients can access or manipulate data and server-updates are administered effectively.

4.3 Monolithic Backend and API Component View

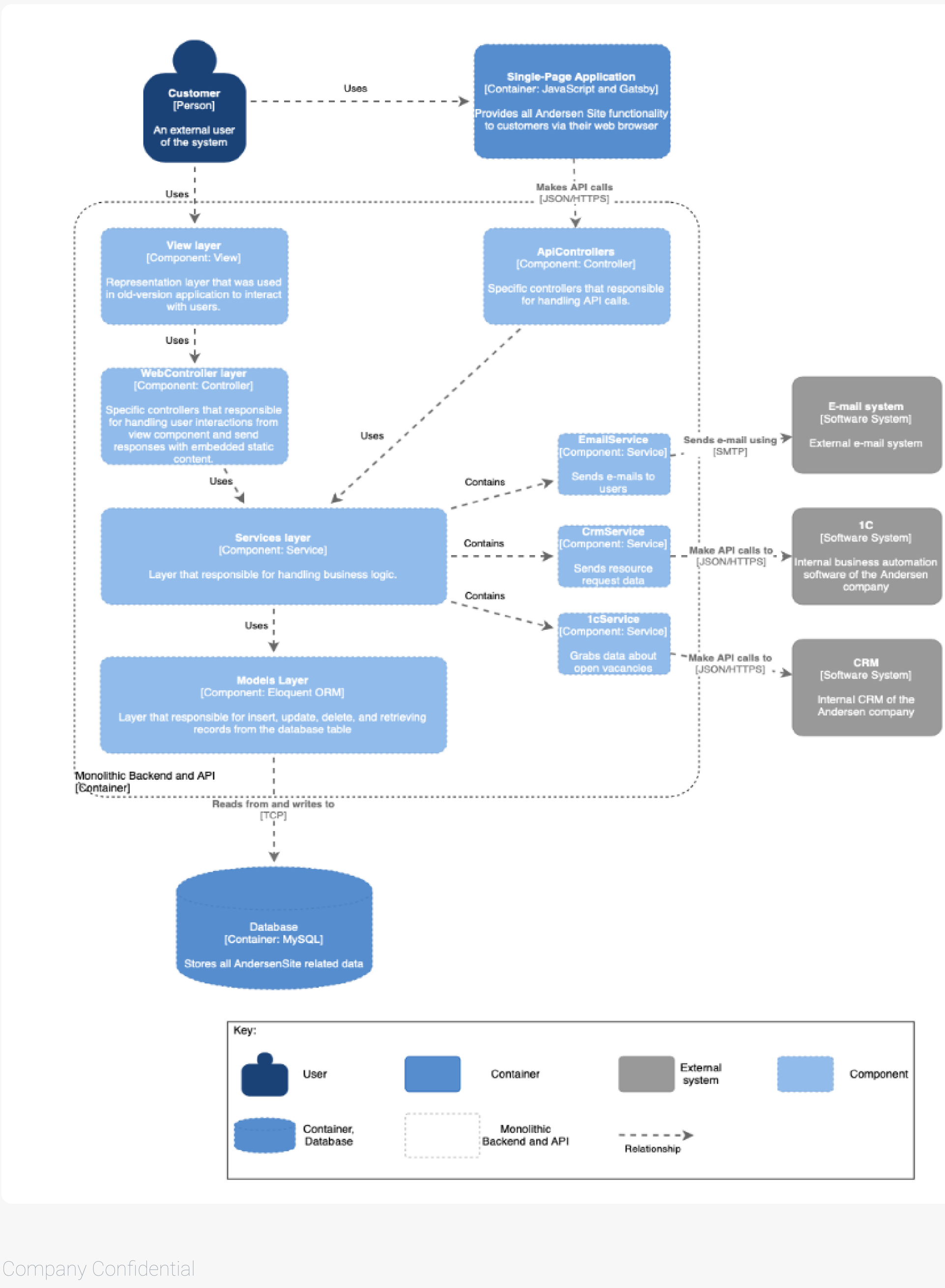
4.3.1 Intent

The view defines the runtime decomposition of the Server-side part of the solution. It is driven by the Business Case and the architecture best practices applicable to the Single-page applications.

4.3.2 Context

The view context is defined by the view Container View where this section represents decomposition of the Monolithic Backend and API component.

4.3.3 Representation



4.3.4 Elements

Table of annotated elements:

Name	Description
Customer	An external user of the system
Single-Page Application	Provides all Andersen Site functionality to customers via their web browser
Monolithic Backend and API	Delivers the static content of the old-version application. Provides new-version AndersenSite functionality via a JSON/HTTPS API. Handle business logic.
View layer	Representation layer used in the old-version application to interact with users.
WebController layer	Specific controllers that are responsible for handling user interactions from the view component and send responses with embedded static content.
ApiControllers	Specific controllers that are responsible for handling API calls.
Services layer	Layer that is responsible for handling business logic.
EmailService	Sends e-mails to users.
CrmService	Sends resources request data.
1cService	Grabs data about open vacancies.

Models Layer	Layer that is responsible for inserting, updating, deleting, and retrieving records from the database table.
Database	Stores all AndersenSite related data.
E-mail system	External e-mail system.
CRM	Internal CRM of the Andersen company.
1C	Internal business automation software of the Andersen company.

4.3.5 Rationale

The section above provides a high-level view of what sort of system needs to be designed, and also gives an opportunity to view the scale of the application by displaying third-party / additional systems and services.

The section Solution Architecture is intended primarily for the Architecture Vision document. It defines and specifies the solution architecture design based on the architecturally significant requirements and constraints identified in the section Architectural Drivers.

5.1 Infrastructure

5.1.1 Infrastructure concepts

According to Design Constraints (CT-5), we decided to use AWS as a cloud provider to host the AndersenSite application.

AWS has the following advantages:

- It is easy to use - AWS is designed to allow application providers, ISVs, and vendors to quickly and securely host your applications – whether an existing application or a new SaaS-based application. You can use the AWS Management Console or well-documented web service APIs to access AWS's application hosting platform.

- It is flexible - AWS enables you to select the operating system, programming language, web application platform, database, and other services you need. With AWS, you receive a virtual environment that enables you to load the software and services your application requires. This eases the migration process for existing applications while preserving options for building new solutions.

- It is reliable - With AWS, you benefit from scalable, reliable, and secure global computing infrastructure, the virtual backbone of Amazon.com's multi-billion dollar online business that has been honed for over a decade.

Operation plan

- It is scalable and high-performing - Using AWS tools, Auto Scaling, and Elastic Load Balancing, your application can be scaled up or down based on demand. Backed by Amazon’s massive infrastructure, you have access to computing and storage resources when you need them.

- It is secure - AWS utilizes an end-to-end approach to secure and strengthen our infrastructure, including physical, operational, and software measures.

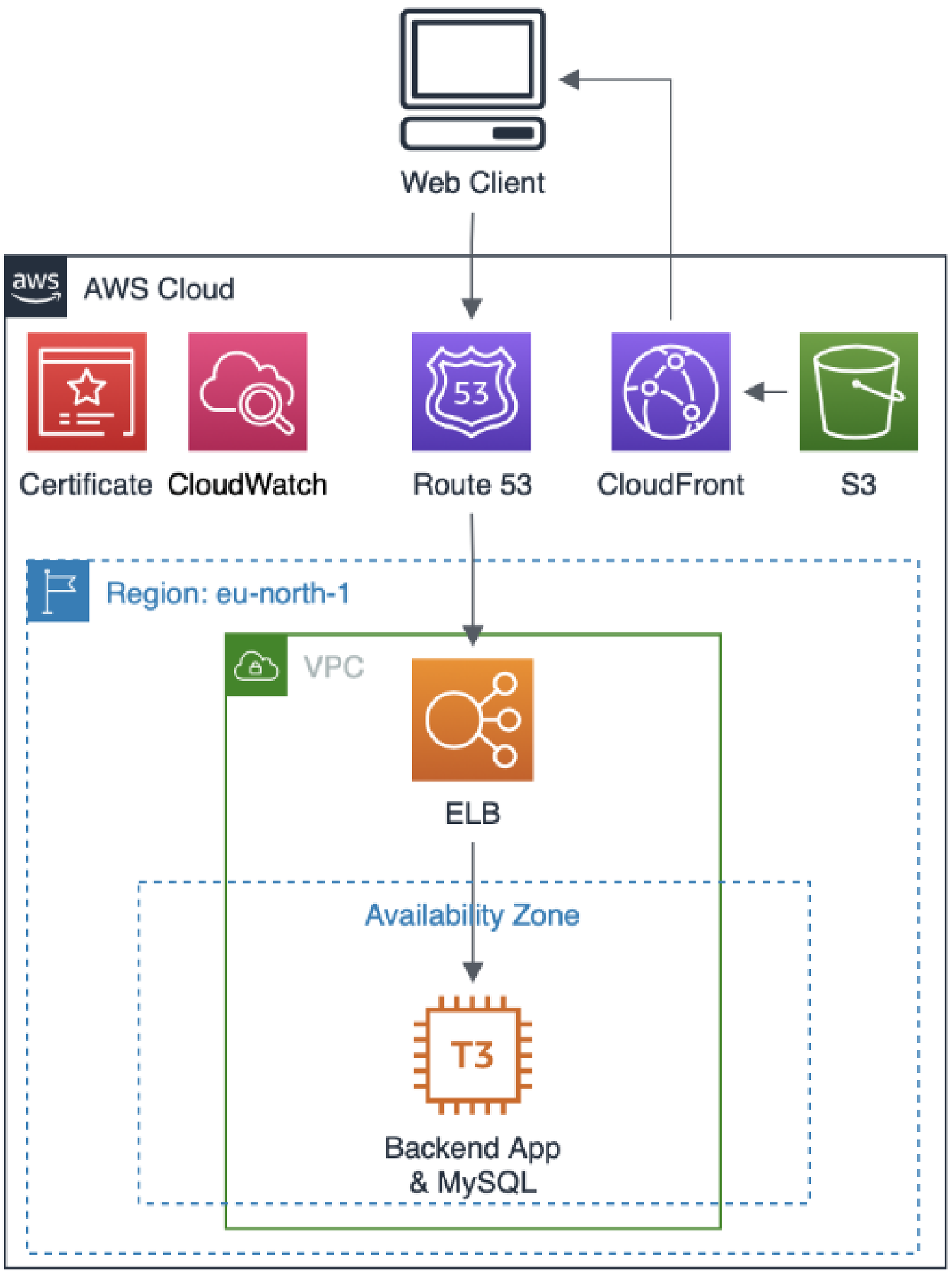
- It is cost-effective - You pay only for the compute power, storage, and other resources you use, with no long-term contracts or up-front commitments.

AWS is made up of multiple different cloud computing products and services. The highly profitable Amazon division provides servers, storage, networking, remote computing, email, mobile development, and security capabilities. AWS can be broken down into the three main products: EC2, Amazon’s virtual machine service, Glacier, a low-cost cloud storage service, and S3, Amazon’s storage system. AWS is so large and present in the computing world that it's far outpaced its competitors.

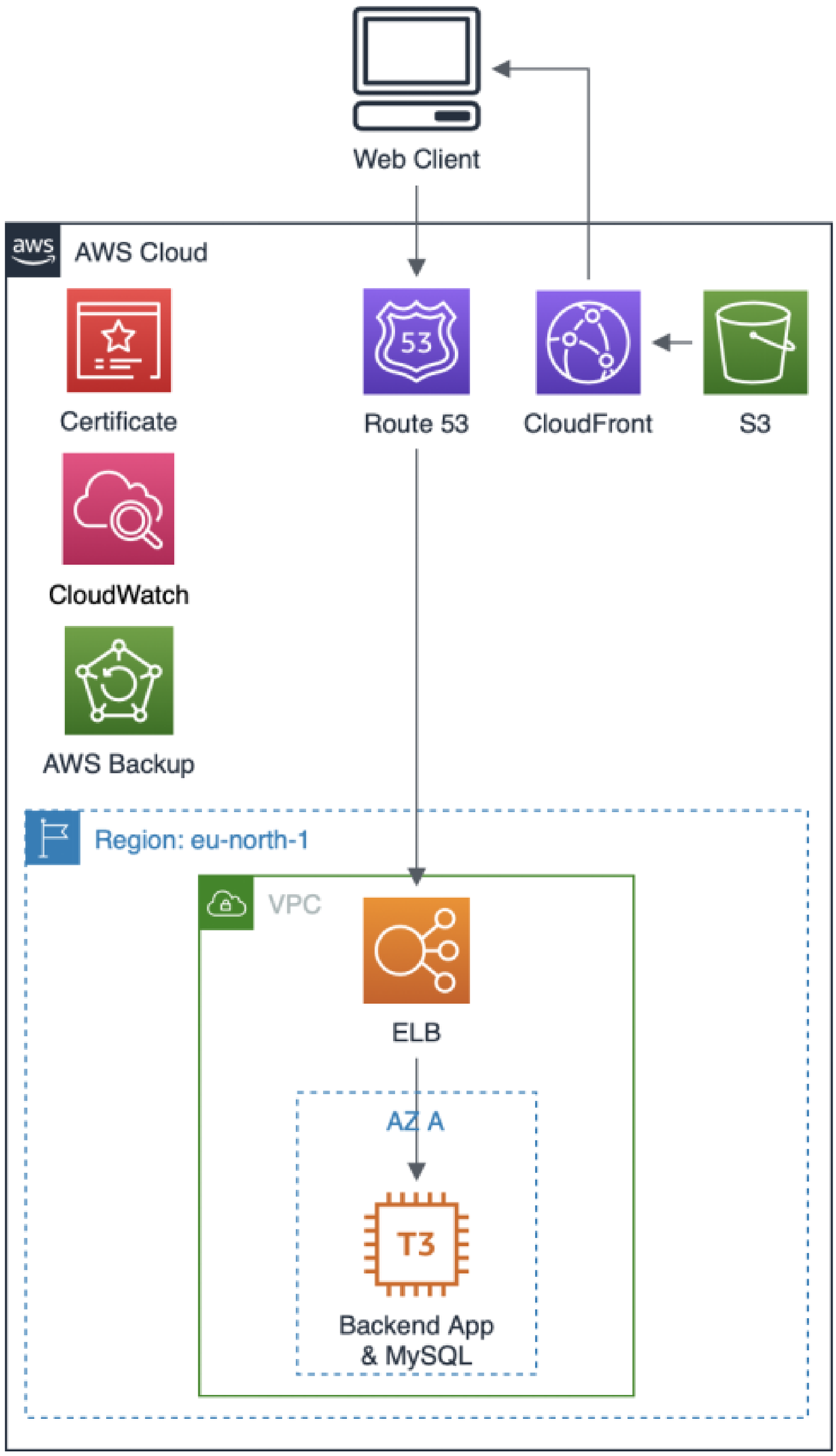
AWS has 76 availability zones in which its servers are located. These serviced regions are divided in order to allow users to set geographical limits on their services (if they choose to do so) and to provide security by diversifying the physical locations where data is kept. Overall, AWS spans across 245 countries and territories.

5.1.2 Environments

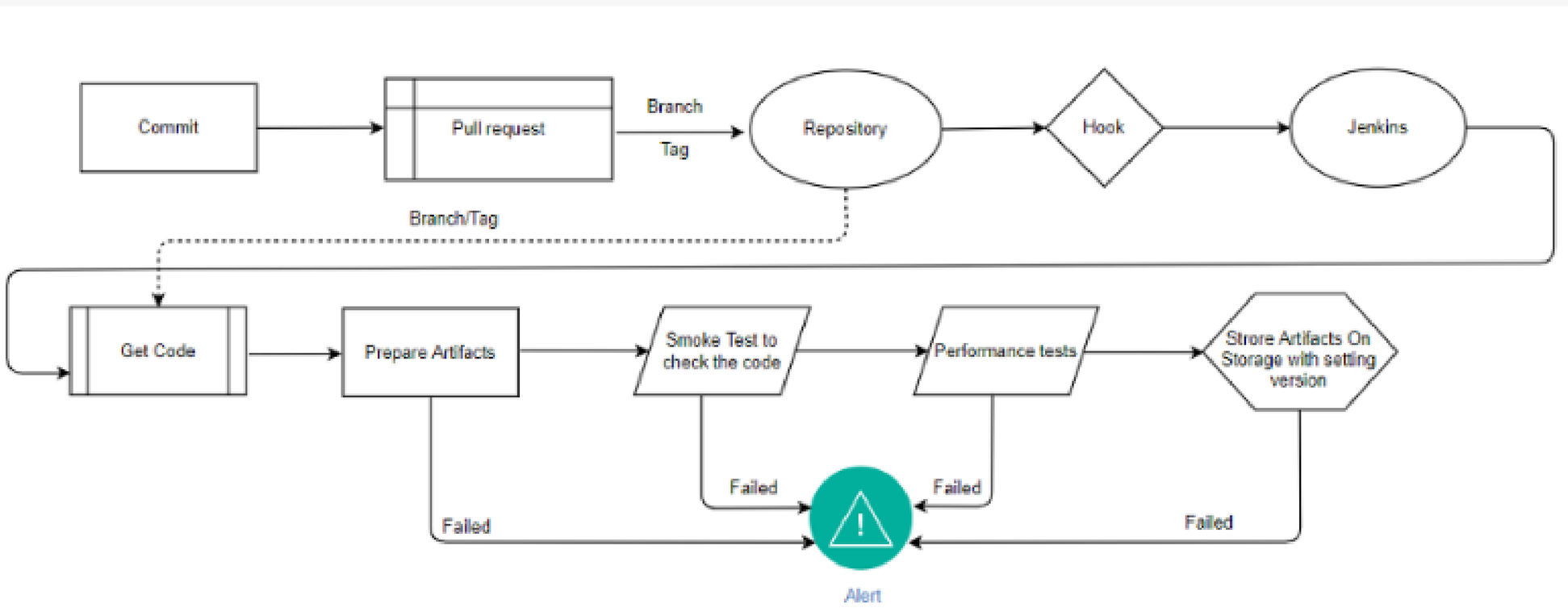
5.1.2.1 Development and Staging environments



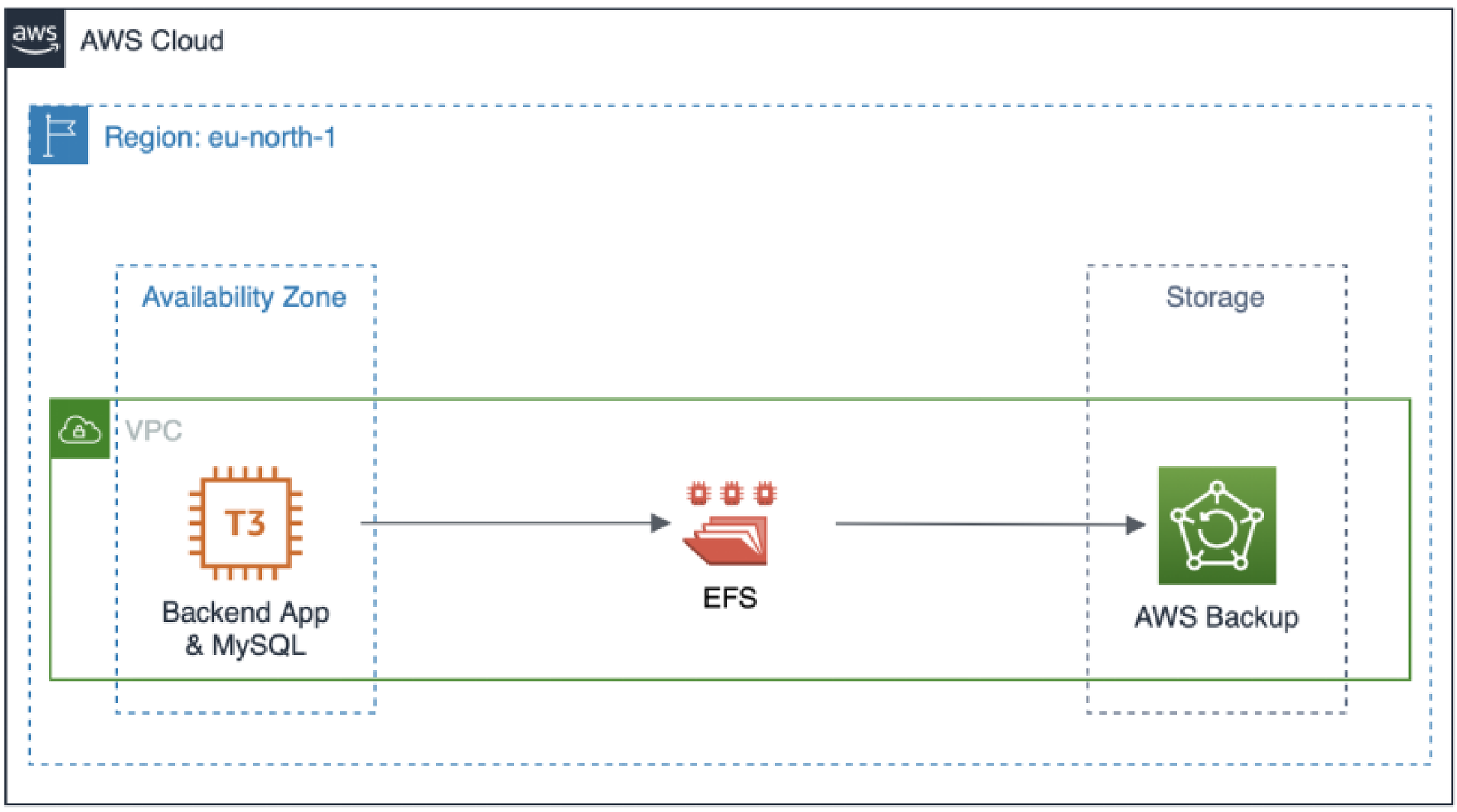
5.1.2.2 Production environment



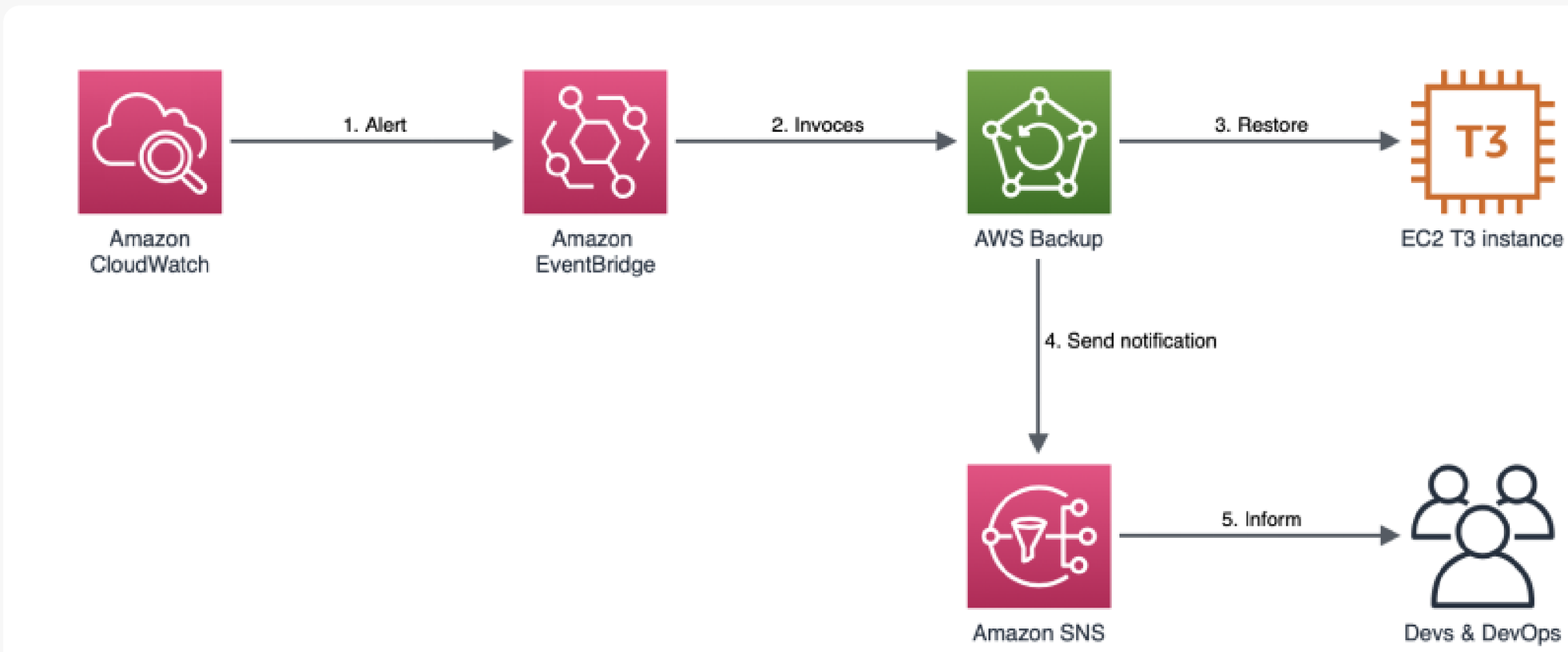
5.2 Continuous integration



5.3 Backups



5.4 Disaster Recovery



6.1 Technology Stack

Technology	Function
Backend	
PHP	Programming language
Laravel	Laravel is a web application framework with expressive, elegant syntax. Laravel attempts to take the pain out of development by easing common tasks used in the majority of web projects, such as authentication, routing, sessions, and caching.
Guzzle	Guzzle is a PHP HTTP client that makes it easy to send HTTP requests and trivial to integrate with web services.
Eloquent ORM	The Eloquent ORM provides an elegant and simple ActiveRecord implementation for working with databases. Each database table has a corresponding "Model" which is used to interact with that table.
MySQL	MySQL is an open-source relational database management system. A flexible and powerful program, MySQL is the most popular open-source database system in the world.
Frontend	
JavaScript	JavaScript (JS) is a lightweight, interpreted, or just-in-time compiled programming language with first-class functions. JavaScript is a prototype-based, multi-paradigm, single-threaded, dynamic language, supporting object-oriented, imperative, and declarative (e.g. functional programming) styles. Read more about JavaScript.

Implementation Roadmap

Gatsby 2.31+	Gatsby enables developers to build fast, secure, and powerful websites using a React-based framework and innovative data layer that makes integrating different content, APIs, and services into one web experience incredibly simple.
MobX 6.0+	MobX is a battle tested library that makes state management simple and scalable by transparently applying functional reactive programming (TFRP).
DevOps	
Docker	Provides the ability to package and run an application in a loosely isolated environment called a container.
Gitlab	GitLab is a web-based Git repository that provides free open and private repositories, issue-following capabilities, and wikis. It is a complete DevOpsplatform that enables professionals to perform all the tasks in a project—from project planning and source code management to monitoring and security.
Jenkins	Jenkins is an open source automation server with an unparalleled plugin ecosystem to support practically every tool as part of your delivery pipelines. Whether your goal is continuous integration, continuous delivery, or something else entirely, Jenkins can help automate it.
Docker Compose	Compose is a tool to define and run multi-container Docker applications. With Compose, you use a YAML file to configure your application's services.

6.2 Estimation

Link to the project estimation

6.3 Team Skillset

Role	Skillset
Solutions Architect	• JavaScript • PHP • Relational DBMS • Cloud experience (AWS/GCP/Azure/etc.) • Jenkins • Terraform • Docker • Technical Debt Management • System analysis and design • ADD
Technical Leader	• JavaScript • PHP • Laravel • React and/or Gatsby • Relational DBMS • Cloud experience (AWS/GCP/Azure/etc.) • Jenkins • Terraform • Docker • Web Security • UML • CI/CD configuration • Docker • Application design skills

Implementation Roadmap

Senior Backend Developer	• PHP • Laravel • Relational DBMS • Web Security • UML • CI/CD configuration • Docker • Application design skills • Cloud experience (AWS/GCP/Azure/etc.)
Senior Frontend Developer	• JavaScript • Web development with React and/or Gatsby • Web Security • UML • CI/CD configuration • Docker • Application design skills • Cloud experience (AWS/GCP/Azure/etc.)
Middle Frontend Developer	• JavaScript • Web development with React and/or Gatsby
DevOps Engineer	• AWS/GCP/Azure/etc. • Jenkins • Terraform • Docker • SonarQube
Business Analyst	• Confluence • Jira • Draw.io (BPMN UML) • Miro • Balsamiq

Implementation Roadmap

Project Manager	• Agile Methodologies (Scrum/Kanban) • Atlassian Jira / Confluence • Asana • MS Project • TeamGantt • Miro • Balsamiq
Manual QA Engineer	• TestRail • Postman • Bug tracking system • DevTools • Jira • Swagger • DBeaver • Allure • Amazon Cloudwatch
Automation QA Engineer	• Java • Maven • Test NG • Rest Assured • Page Object • Selenium WebDriver • Selenium Grid • Allure • JMeter • Fiddler • Git • Jira
UI/UX Designer	• Figma • Miro • Adobe Photoshop • Adobe Color • Jira

6.4 Team Structure

Role	Skillset	Count	FTE
Solutions Architect	• System analysis and design • System requirements specification • Documenting and monitoring requirements needed to implement proposed requirements • Provision of detailed specifications for proposed solutions • General Technical Leadership • Negotiation with customers	1	0.25
Technical Leader	• Technical team leadership • Technical communication • Code review • Backend implementation • Pipeline implementation • Delegating tasks and achieving daily, sprint, and release goals. • Motivating staff and creating a space where they can ask questions and voice their concerns. • Being transparent with the team about challenges, failures, and successes. • Writing progress reports and delivering presentations to the relevant stakeholders.	1	0.5
Senior Backend Engineer	• Technical communication • Code review • Backend implementation • Being involved and participating in the overall application lifecycle • Main focus on coding and debugging	1	1

Implementation Roadmap

	• Collaboration with Frontend developers • Provision of training, help and support to other team members • Building high-quality reusable code that can be used in the future • Developing functional and sustainable web applications with clean codes • Troubleshooting and debugging the applications		
Senior Frontend Engineer	• Technical communication • Code review • Frontend implementation • Being involved and participating in the overall application lifecycle • Main focus on coding and debugging • Collaboration with Backend developers • Provision of training, help and support to other team members • Building high-quality reusable code that can be used in the future • Developing functional and sustainable web applications with clean codes • Troubleshooting and debugging the applications	1	1
Middle Frontend Engineer	• Frontend implementation • Compiling and analyzing data, processes, and codes to troubleshoot problems and identify areas for improvement. • Collaborating with the front-end developers and other team members to establish objectives and design more functional, cohesive codes to enhance the user experience. • Developing ideas for new programs, products, or features by monitoring industry developments and trends. • Participating in continuing education and training to remain current on best practices, learn new programming languages, and better assist other team members	2	2

Implementation Roadmap

QA Lead	• Managing the QA team • Overseeing the drafting of testing documents. • Implementing testing procedures and overseeing the QA process. • Troubleshooting quality issues and modifying test procedures. • Conducting analysis checks on product specifications. • Reviewing Quality Assurance reports. • Ensuring the successful deployment of products into the market. • Responding to requests from the design team and management. • Implementing testing procedures and overseeing the QA process.	1	0.5
QA Engineer	• Planing, executing, and overseeing inspection and testing of incoming and outgoing products to confirm quality conformance to specifications and quality deliverables • Analyzing and investigating product complaints or reported quality issues to ensure closure in accordance with company guidelines and external regulatory requirements	2	2
Automation QA Engineer	• Planing, executing, and overseeing inspection and testing of incoming and outgoing product to confirm quality conformance to specifications and quality deliverables • Analyzing and investigating product complaints or reported quality issues to ensure closure in accordance with company guidelines and external regulatory requirements	1	0.5
DevOps Engineer	• Cloud infrastructure setup • Deployment setup • Build pipeline review	1	0.25

Implementation Roadmap

Business Analyst Lead	• Managing BA team • Understanding what business does and how it does • Determining how to improve existing business processes • Identifying the steps or tasks to support the implementation of new features • Designing the new features to implement • Analyzing the impact of implementing new features	1	0.5
Business Analyst	• Identifying the steps or tasks to support the implementation of new features • Designing the new features to implement • Analyzing the impact of implementing new features	2	2
UI/UX Engineer	• Investigating user experience design requirements for our suite of digital assets • Developing and conceptualizing a comprehensive UI/UX design strategy for the brand • Producing high quality UX design solutions through wireframes, visual and graphic designs, flow diagrams, storyboards, site maps, and prototypes • Designing UI elements and tools such as navigation menus, search boxes, tabs, and widgets for our digital assets • Testing UI elements such as CTAs, banners, page layouts, page designs, page flows, and target links for landing pages • Providing advice and guidance on the implementation of UX research methodologies and testing activities in order to analyze and predict user behavior • Adhering to style standards on typography and graphic design	2	2
Project Manager	• Designing and applying appropriate project management standards • Managing the production of the required deliverables	1	0.5

Implementation Roadmap

	• Planning and monitoring the project • Preparing and maintaining project, stage and exception plans as required • Managing project risks, including the development of contingency plans • Monitoring overall progress and use of resources, initiating corrective action where necessary		
Delivery Manager	• Maintaining end-to-end accountability for customer satisfaction and overall delivery excellence within specific services • Ensure that the right type and number of resources that are required to fulfill the planned projects are available and in place • Delivering customer satisfaction and overall excellence by identifying opportunities (or issues) and assisting with speedy resolution • Responsible for financial management and reporting and optimizing processes • Continuously improving the technical delivery model and strategy, implementing and managing delivery with the associated teams	1	0.1