Architecture Document

Memi Lavi
www.memilavi.com

Architecture Document

- Contains:
  - The architecture (surprise!)
  - Functional & Non-Functional requirements
  - Technology stack
  - And more…

No Development

Before Document!

Get Your

Architecture Document!

Goals of the Architecture Document

- Describe what should be developed and how
- Lay out the requirements (functional & non-functional)

Audience of the Architecture Document

- Almost everyone involved
    - Project Manager
    - CTO
    - QA Leader
    - Developers (of course…)

Development Team

- The document lays out the basic concepts of the system
  - Technology Stack
  - Components
  - Services
  - Communication
  - And more…

Management

- Project Manager, CTO, CEO
- “The Project Is in Good Hands”
    - Requirements reflect the essence of the system
    - Executive Summary describes best practices and modern patterns
    - Architecture geared towards business goals
- Management’s sections appear first

QA Lead

- Prepare testing infrastructure
  - Servers
  - Testing tools
  - Coding

Format of the Architecture Document

- Subject to hot debates
- There are some standards
- UML – One of the most famous

UML

- Modeling Language
- Visualizes system’s design
- Consists of Concepts and Diagrams

UML – contd.

- I don’t use it
- The audience is usually not familiar with it
- Requires a lot of time for explaining

Recommended Format

- As Simple As Possible
- Use plain English
- Get into the minds of your readers
- Visualize using software you're comfortable with

Background

Requirements

Executive Summary

Architecture Overview

Components Drill-Down

Background

Length : <= One Pager

Audience: Team & Management

Background

- Describes the system from a Business POV
  - System’s role
  - Reasons for replacing the old system
  - Expected business impact

Background

?

Background

- Because:
  - Validate your point of view
  - Boost confidence in you

Background

Length :	<= One Pager
Audience:	Team & Management

Requirements

Length :	<= One Pager
Audience:	Team & Management

Requirements

Requirement Type	Description
Functional	What the system should do?

Requirements

Requirement Type	Description
Functional	What the system should do?
Non-Functional	What should the system deal with?

Requirements

Requirement Type	Description
Functional	What the system should do?
Non-Functional	With what should the system deal? (Performance, Load, Data Volume, SLA and more)

Requirements

- Brief
- Bulleted List
- No more than 3 lines per requirement

Requirements

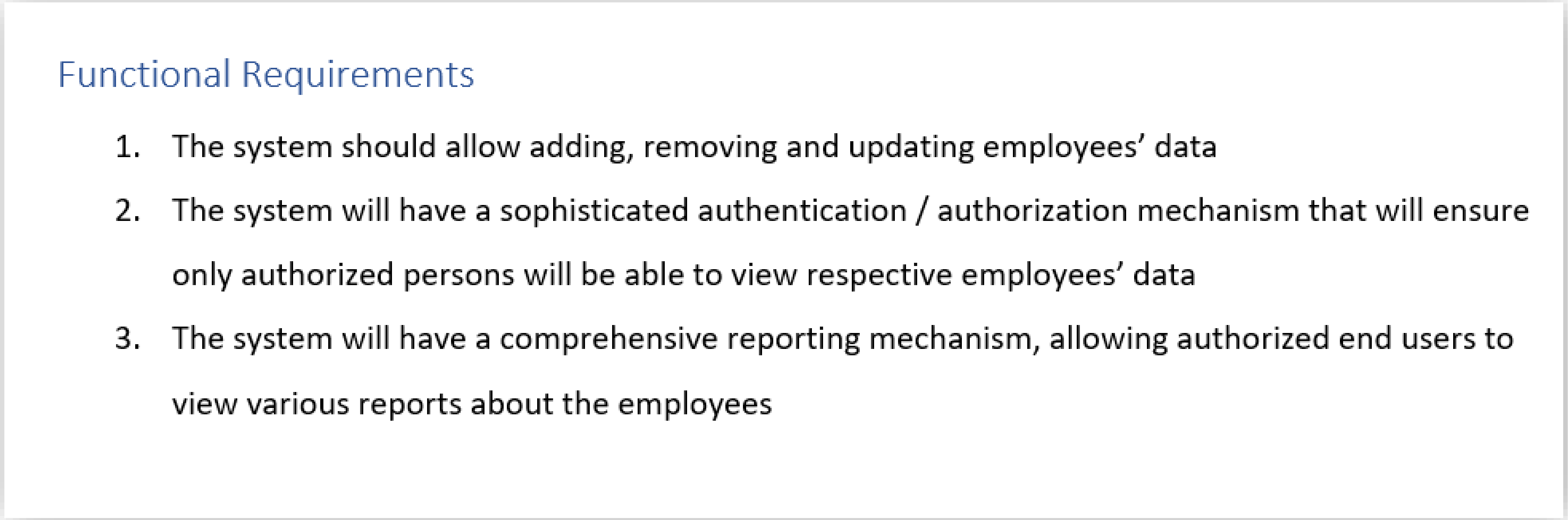
?

Requirements

- Because:
  - Validate your understanding of the requirements
  - Requirements dictate Architecture

Requirements

- Structure:
  - First – Functional Requirements

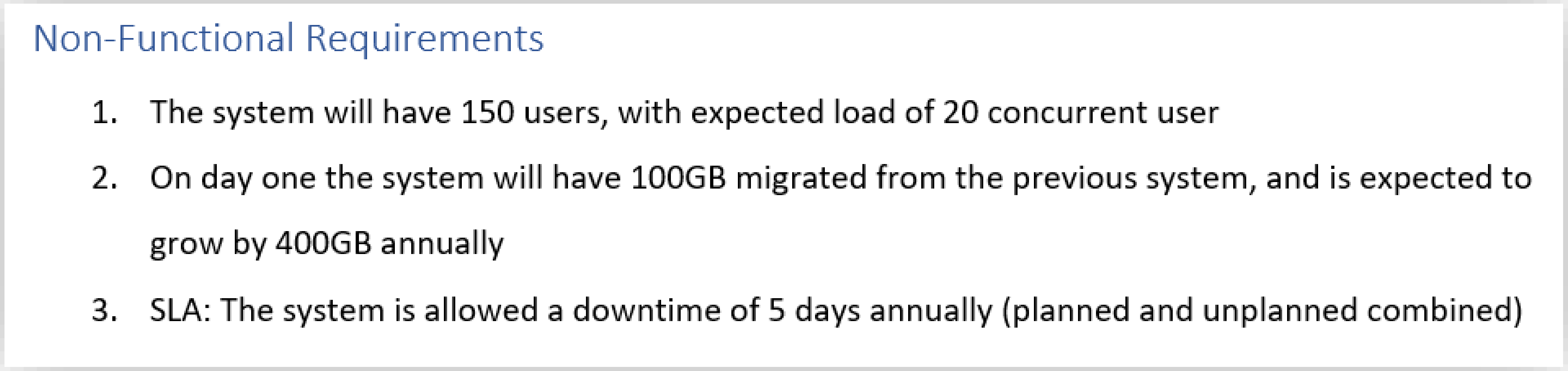


Functional Requirements

1. The system should allow adding, removing and updating employees' data
2. The system will have a sophisticated authentication / authorization mechanism that will ensure only authorized persons will be able to view respective employees' data
3. The system will have a comprehensive reporting mechanism, allowing authorized end users to view various reports about the employees

Requirements

- Structure:
  - Next – Non-Functional Requirements
  - Should be extremely accurate and specific



Non-Functional Requirements

1. The system will have 150 users, with expected load of 20 concurrent user
2. On day one the system will have 100GB migrated from the previous system, and is expected to grow by 400GB annually
3. SLA: The system is allowed a downtime of 5 days annually (planned and unplanned combined)

Background	Length : <= One Pager Audience: Team & Management
Requirements	Length : <= One Pager Audience: Team & Management
Executive Summary	Length : <= 3 Pages Audience: Management

Executive Summary

?

Executive Summary

- Management is busy, non-technical
- Managers will NOT read the whole document

Executive Summary

- Goal:
    - Provide high-level view of the architecture
    - Boost confidence of your work
- Get into your readers' mind!

Executive Summary

- Tips:
  - Use charts and diagrams
  - Write it AFTER writing the rest of the document
  - Use well known technical terms – sparsely!
  - Do not repeat yourself

Background	Length : <= One Pager	Audience: Team & Management
Requirements	Length : <= One Pager	Audience: Team & Management
Executive Summary	Length : <= 3 Pages	Audience: Management
Architecture Overview	Length : <= 10 Pages	Audience: Development Team & QA Lead

Architecture Overview

- Provides high-level view of the architecture
- Presents the architecture to the team
- No deep-dive to specific components

General Description

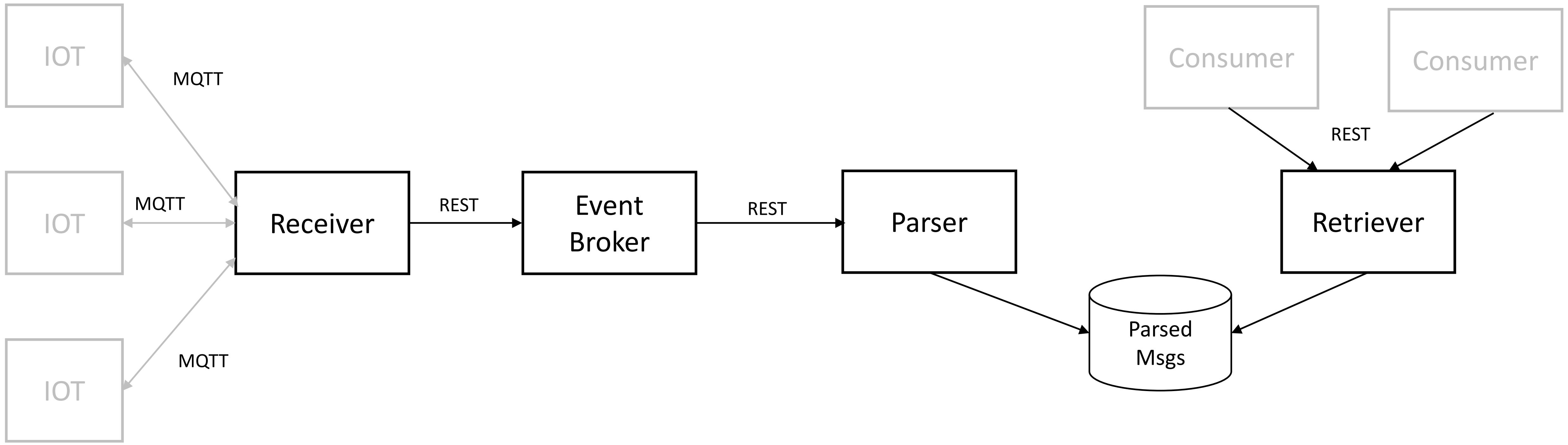
Type	Web-Based, Micro Services, REST API
Major NF-Requirements	50 Reqs / Sec

General Description

Type	Web-Bases, Micro Services, REST API
Major NF-Requirements	50 Reqs / Sec

High-Level Diagram

No formal visualization standard



Logic Diagram Only!

General Description

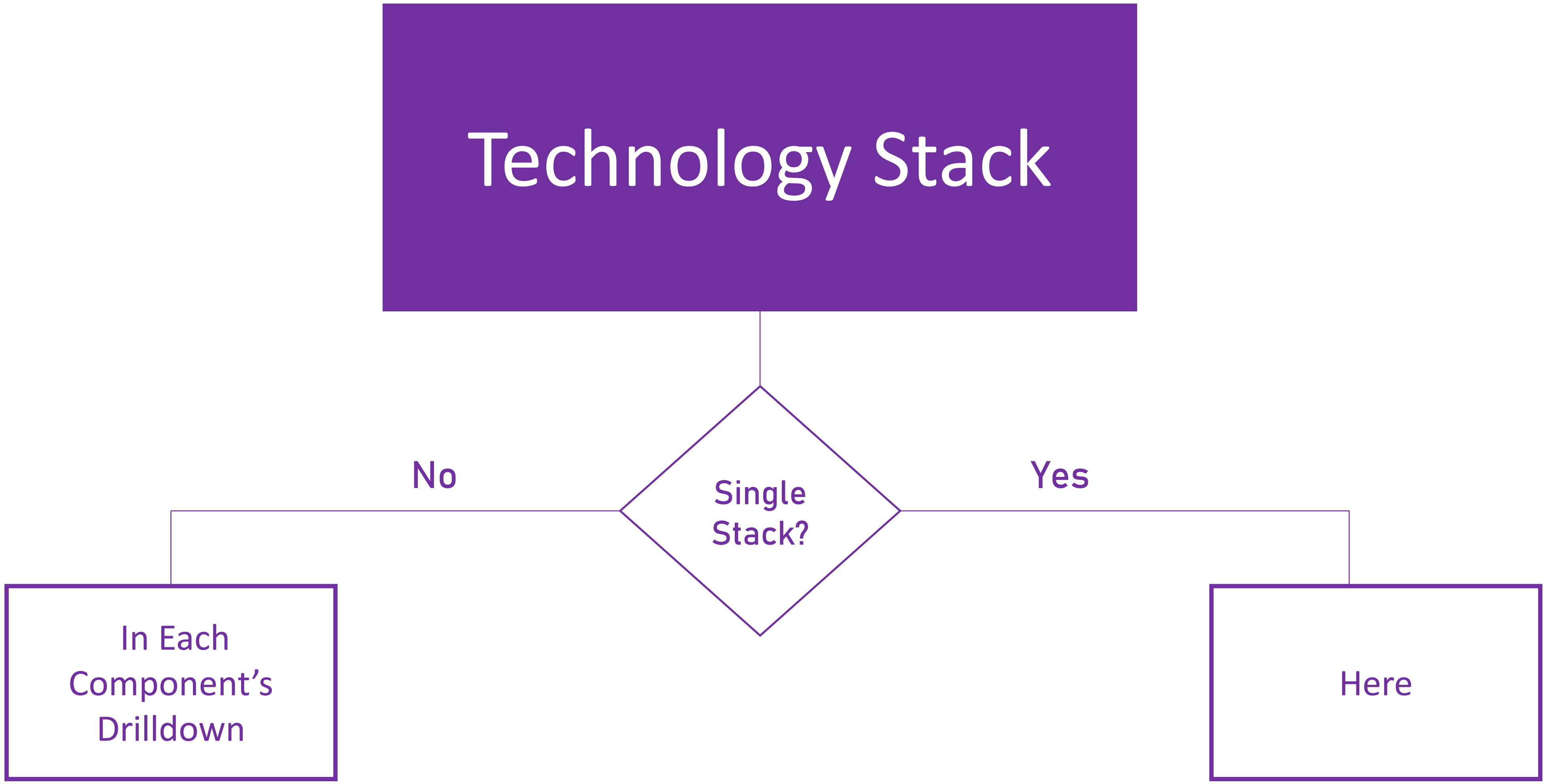
Type	Web-Bases, Micro Services, REST API
Major NF-Requirements	50 Reqs / Sec

High-Level Diagram

No formal visualization standard

Diagram Walkthrough

- Describes various parts and their role
- Uses simple words
- Includes all relevant details


Technology Stack
Single Stack?
No
Yes
In Each Component's Drilldown
Here

Background	Length : <= One Pager	Audience: Team & Management
Requirements	Length : <= One Pager	Audience: Team & Management
Executive Summary	Length : <= 3 Pages	Audience: Management
Architecture Overview	Length : <= 10 Pages	Audience: Development Team & QA Lead
Components Drill-Down	Length : Unlimited	Audience: Development Team & QA Lead

Components Drill Down

For Each Component:

Component's Role	Recap of the component's description from the Architecture Overview section
Technology Stack	Technologies used in developing the component

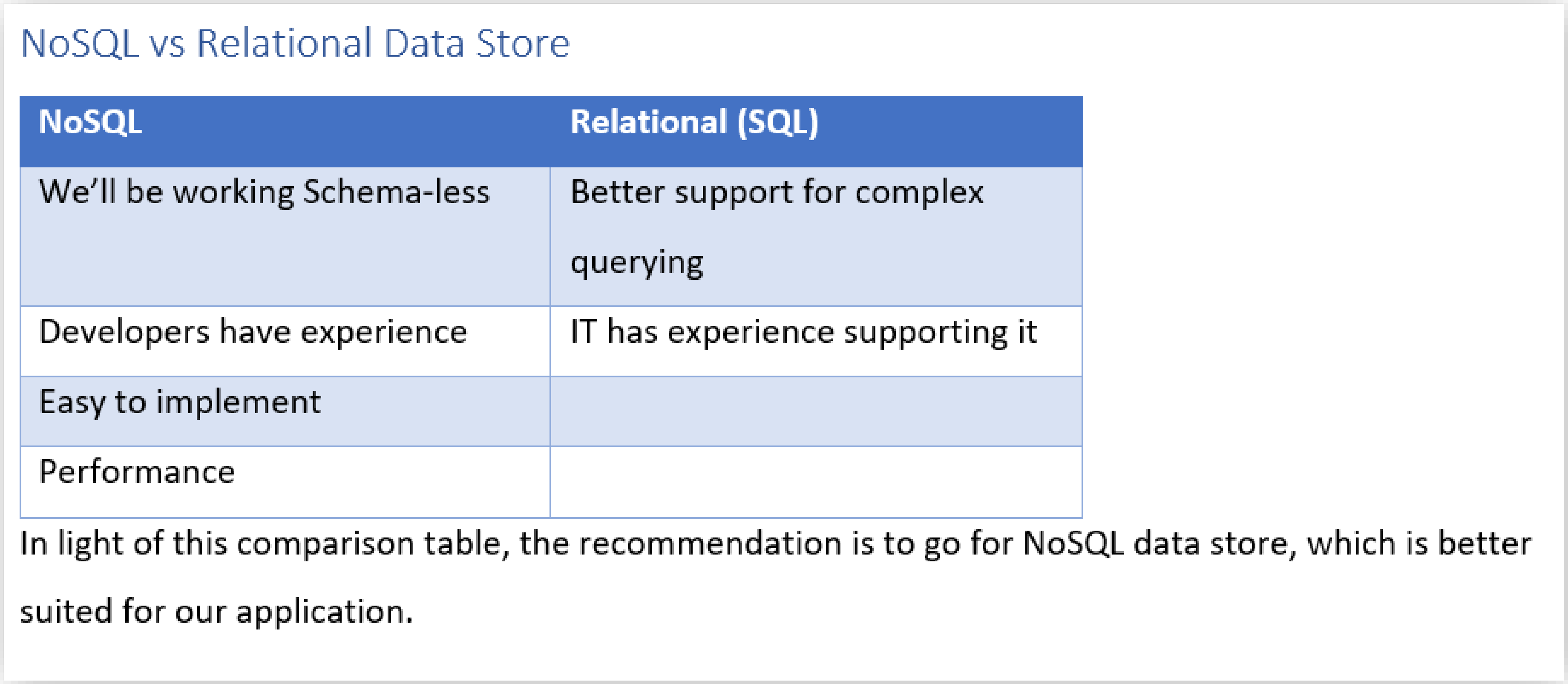
Technology Stack

- Lay out the various parts that need to be specified
  - Data Store
  - Back End
  - Front End
  - Etc…

CAUTION!

Be Extremely Detailed!

Always Include Rationale!



NoSQL vs Relational Data Store

NoSQL	Relational (SQL)
We'll be working Schema-less	Better support for complex querying
Developers have experience	IT has experience supporting it
Easy to implement	
Performance	

In light of this comparison table, the recommendation is to go for NoSQL data store, which is better suited for our application.

No need to repeat for every component

Components Drill Down

For Each Component:

- **Component's Role** — Recap of the component's description from the Architecture Overview section
- **Technology Stack** — Technologies used in developing the component
- **Component's Architecture** — The inner architecture of the component

Inner Architecture

- Describe the API
  - Include method names!

URL	Role	Response Code	Comments
GET /api/employees/*id*	Retrieve employee by ID	200 (OK) 404 (Not Found)	
POST /api/employees	Add new employee	201 (Created) 400 (Bad Request)	Return 400 if any major field (such as birth date) is invalid
GET /api/employees/*id* /division	Retrieve specific employee's division	200 (OK) 404 (Not Found)	

Inner Architecture

- Describe the layers
- Include important considerations (DI, etc…)
- Be as detailed as possible

Components Drill Down

For Each Component:

Component's Role	Recap of the component's description from the Architecture Overview section
Technology Stack	Technologies used in developing the component
Component's Architecture	The inner architecture of the component
Development Instructions	Specific development guidelines

Your Architecture Document Waits at the End of the Next Section!

Architecture Document

- The center of the Architect’s work
- MUST include all insights
- Get into your readers’ mind