Requirement Gathering

Hawaii Annual Coding Challenge 2022

Presenter Name: Joel Kumabe

Joel Kumabe

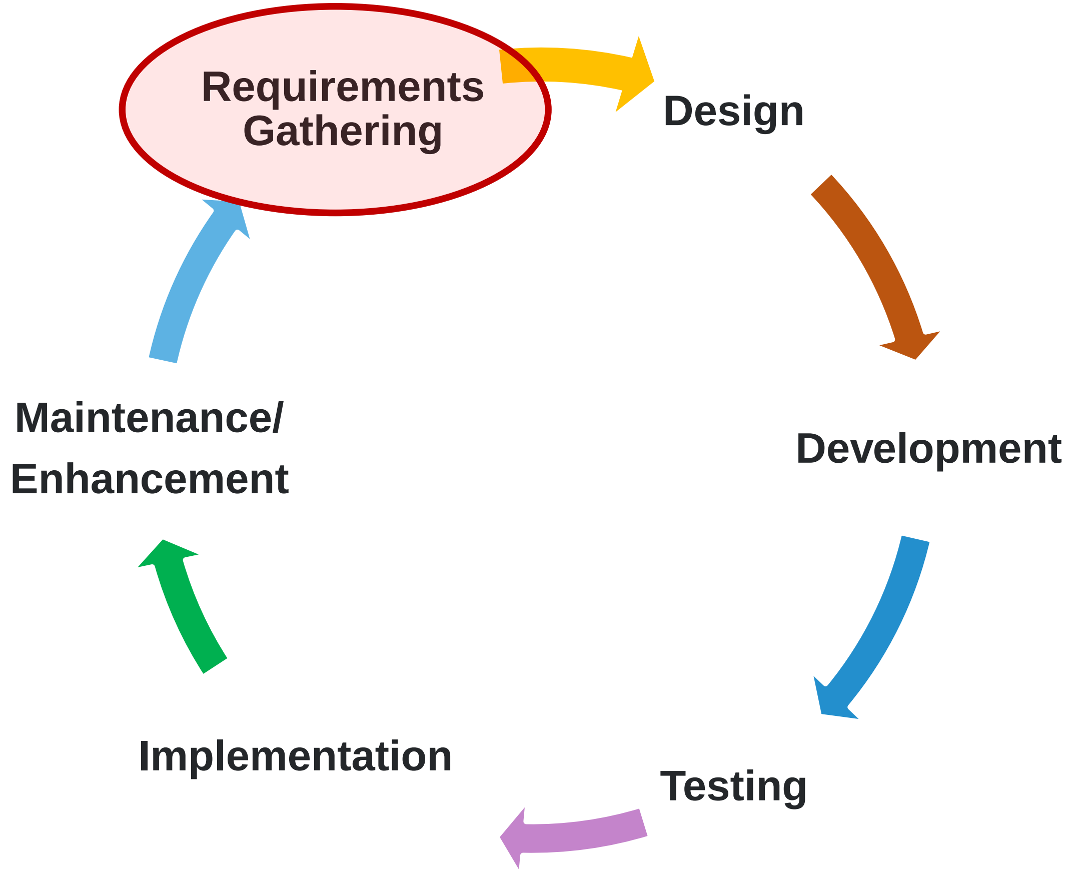
Joel Kumabe is the Client Executive in charge of Hawaii for Unisys Corporation.

Mr. Kumabe has over 38+ years of experience in technology, primarily in Banking and Healthcare Insurance.

Prior to Unisys, Mr. Kumabe was the Senior Vice-President and Chief Information Officer at Hawaii State Federal Credit Union, Program Manager at HMSA, Enterprise Architect at Kamehameha Schools/Bishop Estate, Vice-President and Application Manager at Bank of Hawaii, and Consulting Manager at Ernst & Young, CPAs.

|

Software Development Lifecycle



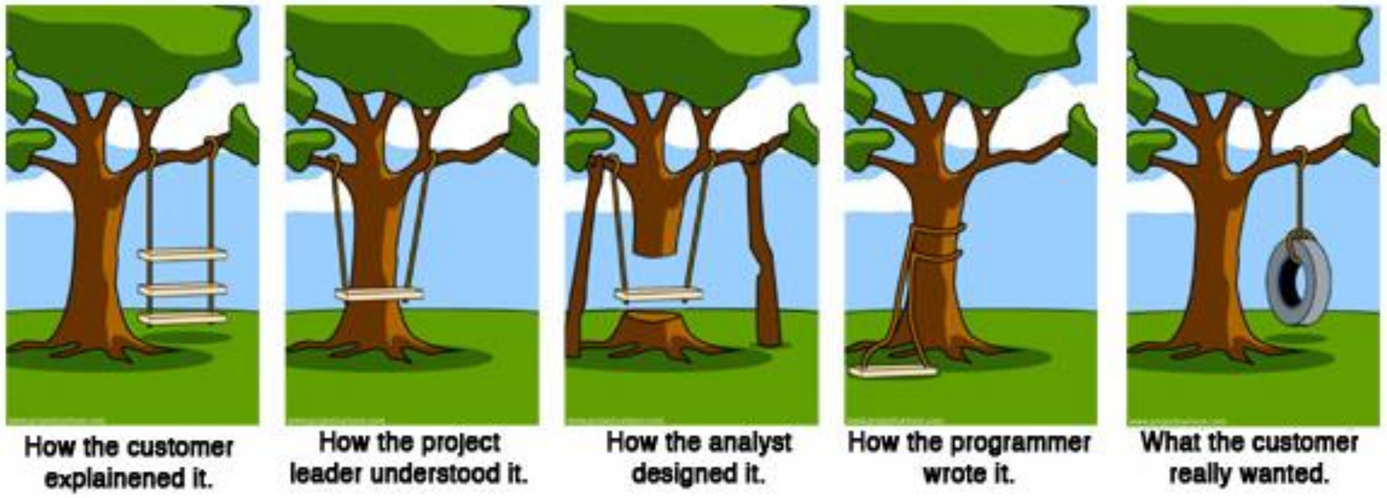
What is Requirements Gathering?

This introduction to requirements gathering is meant to help you understand the methods, planning and processes that will lead to designing and developing systems that meet user's needs.

Requirements gathering is **the process of determining what your projects need to achieve and what needs to be created to make that happen**. You're probably familiar with the fact that everybody has their own common project assumptions about what a project should include.

Requirements are a specification of what should be implemented. They are descriptions of how the system should behave, and/or of a system property or attribute. They may also include constraints on the development process of the system.

Why Requirements Gathering?


How the customer explainened it.
How the project leader understood it.
How the analyst designed it.
How the programmer wrote it.
What the customer really wanted.

Why Requirements Gathering?

Requirements are needed to insure that what is expected is actually designed, developed, tested and approved.

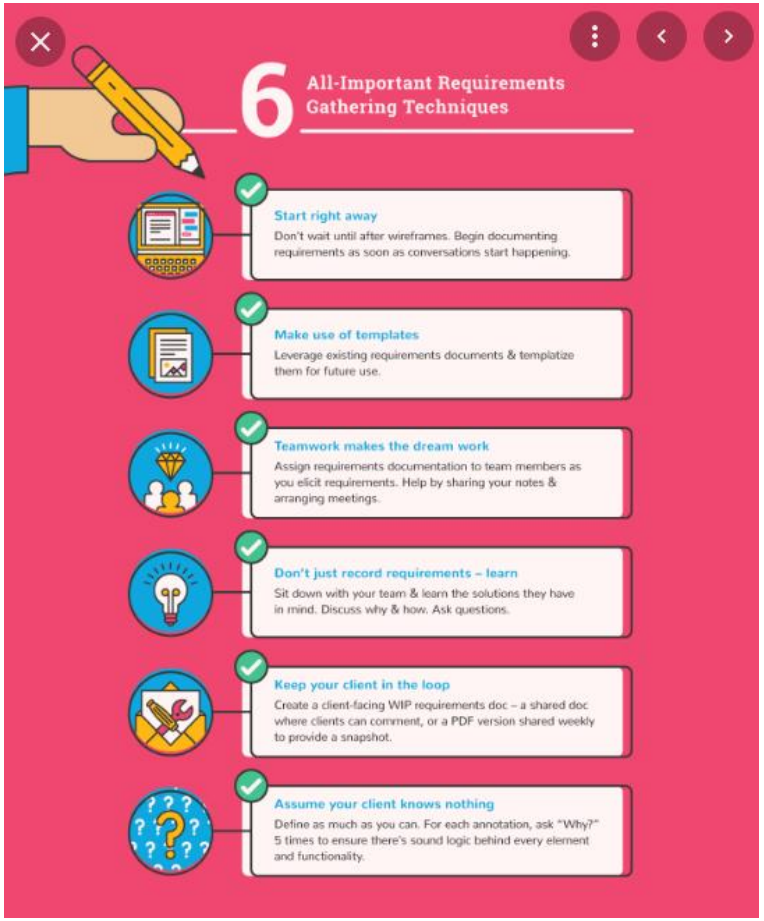
Good requirements can:

- Increased customer and client satisfaction – fewer missed expectations
- Faster product development and delivery – fewer delays
- Increase Consistency and re-use
- Reduced need to rework
- Reduce cost
- Increased trust
- Provide effective communications
- Fewer defects
- Prevents “Scope Creep”

Types of Requirements Gathering Techniques

- Interviews
- Questionnaires or Surveys
- User Observation
- Document Analysis
- Interface analysis
- Workshops
- Brainstorming
- Role-play
- Use Cases and Scenarios
- Focus Groups
- Prototyping

Requirements Gathering Tips



6 All-Important Requirements Gathering Techniques

Start right away
Don't wait until after wireframes. Begin documenting requirements as soon as conversations start happening.

Make use of templates
Leverage existing requirements documents & templatize them for future use.

Teamwork makes the dream work
Assign requirements documentation to team members as you elicit requirements. Help by sharing your notes & arranging meetings.

Don't just record requirements – learn
Sit down with your team & learn the solutions they have in mind. Discuss why & how. Ask questions.

Keep your client in the loop
Create a client-facing WIP requirements doc – a shared doc where clients can comment, or a PDF version shared weekly to provide a snapshot.

Assume your client knows nothing
Define as much as you can. For each annotation, ask "Why?" 5 times to ensure there's sound logic behind every element and functionality.

Keep an open mind

Actively Listen

Gather what's requested and avoid trying to develop a solution

Software Development Methodologies

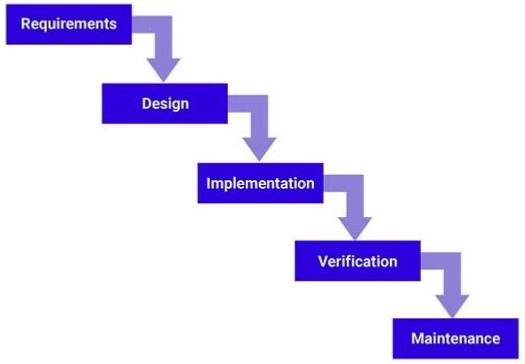
- Agile development methodology.
- DevOps deployment methodology.
- Waterfall development method.
- Rapid application development.

Top 4 software development methodologies
Synopsis.com

Software Development Lifecycle

Waterfall development method

Many consider the waterfall method to be the most traditional software development method. The waterfall method is a rigid linear model that consists of sequential phases (requirements, design, implementation, verification, maintenance) focusing on distinct goals. Each phase must be 100% complete before the next phase can start. There's usually no process for going back to modify the project or direction.



Software Development Lifecycle

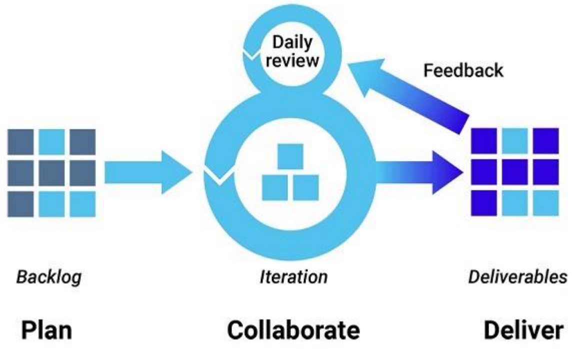
Pros: The linear nature of the waterfall development method makes it easy to understand and manage. Projects with clear objectives and stable requirements can best use the waterfall method. Less experienced project managers and project teams, as well as teams whose composition changes frequently, may benefit the most from using the waterfall development methodology.

Cons: The waterfall development method is often slow and costly due to its rigid structure and tight controls. These drawbacks can lead waterfall method users to explore other software development methodologies.

Software Development Lifecycle

Agile development methodology

Teams use the agile development methodology to minimize risk (such as bugs, cost overruns, and changing requirements) when adding new functionality. In all agile methods, teams develop the software in iterations that contain mini-increments of the new functionality. There are many different forms of the agile development method, including scrum, crystal, extreme programming (XP), and feature-driven development (FDD).



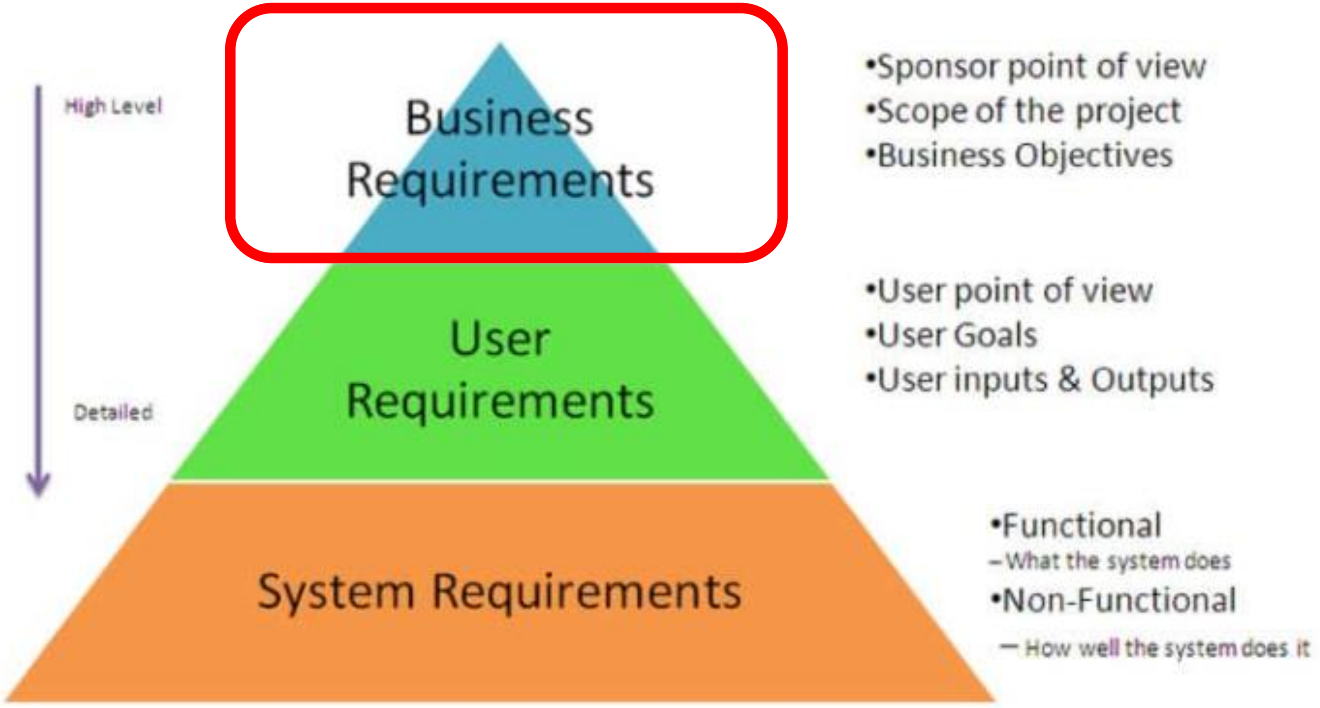
Software Development Lifecycle

Pros: The primary benefit of agile software development is that it allows software to be released in iterations. Iterative releases improve efficiency by allowing teams to find and fix defects and align expectation early on. They also allow users to realize software benefits earlier, with frequent incremental improvements.

Cons: Agile development methods rely on real-time communication, so new users often lack the documentation they need to get up to speed. They require a huge time commitment from users and are labor intensive because developers must fully complete each feature within each iteration for user approval.
Agile development methods are similar to rapid application development (see below) and can be inefficient in large organizations. Programmers, managers, and organizations accustomed to the waterfall method (see below) may have difficulty adjusting to an agile SDLC. So a hybrid approach often works well for them.

UNISYS | Securing Your Tomorrow®

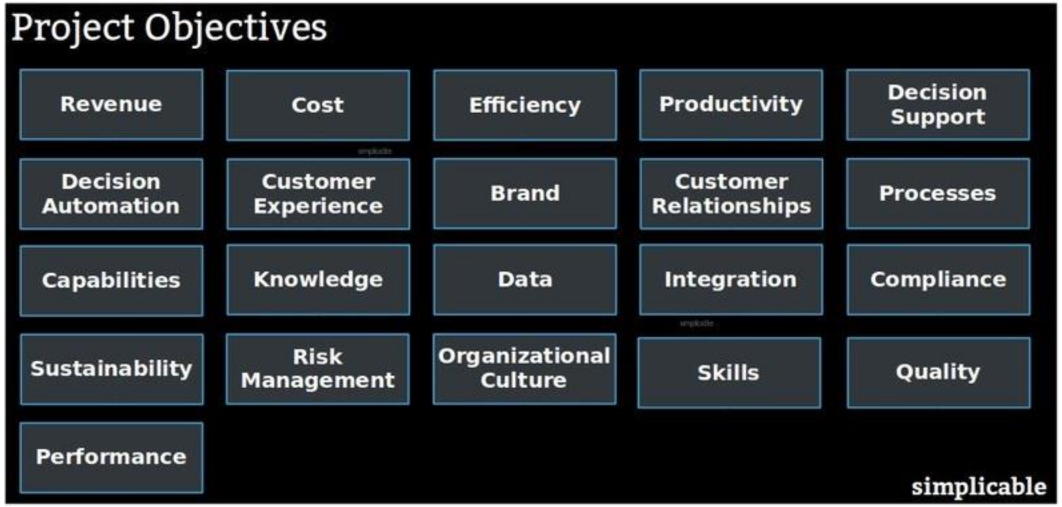
Types of Requirements Gathering



Note: Inverse relationship to Project Impact

How to Start - Project Objective

Statement outlining the overall objective of the project. Meaningful steps towards meeting a business goal. Used to communicate project purpose, direction, value, and progress:



Project Objectives

Revenue	Cost	Efficiency	Productivity	Decision Support
Decision Automation	Customer Experience	Brand	Customer Relationships	Processes
Capabilities	Knowledge	Data	Integration	Compliance
Sustainability	Risk Management	Organizational Culture	Skills	Quality
Performance				

simplicable

Who, What, Why, When, How

First Question: What are we doing? What is the scope of the project?

Who is the sponsor?   Who are the end users? Who makes decisions?

Determine if it is truly a requirement or something that is not absolutely needed -

How much money, people, materials, machine, etc. do you have?

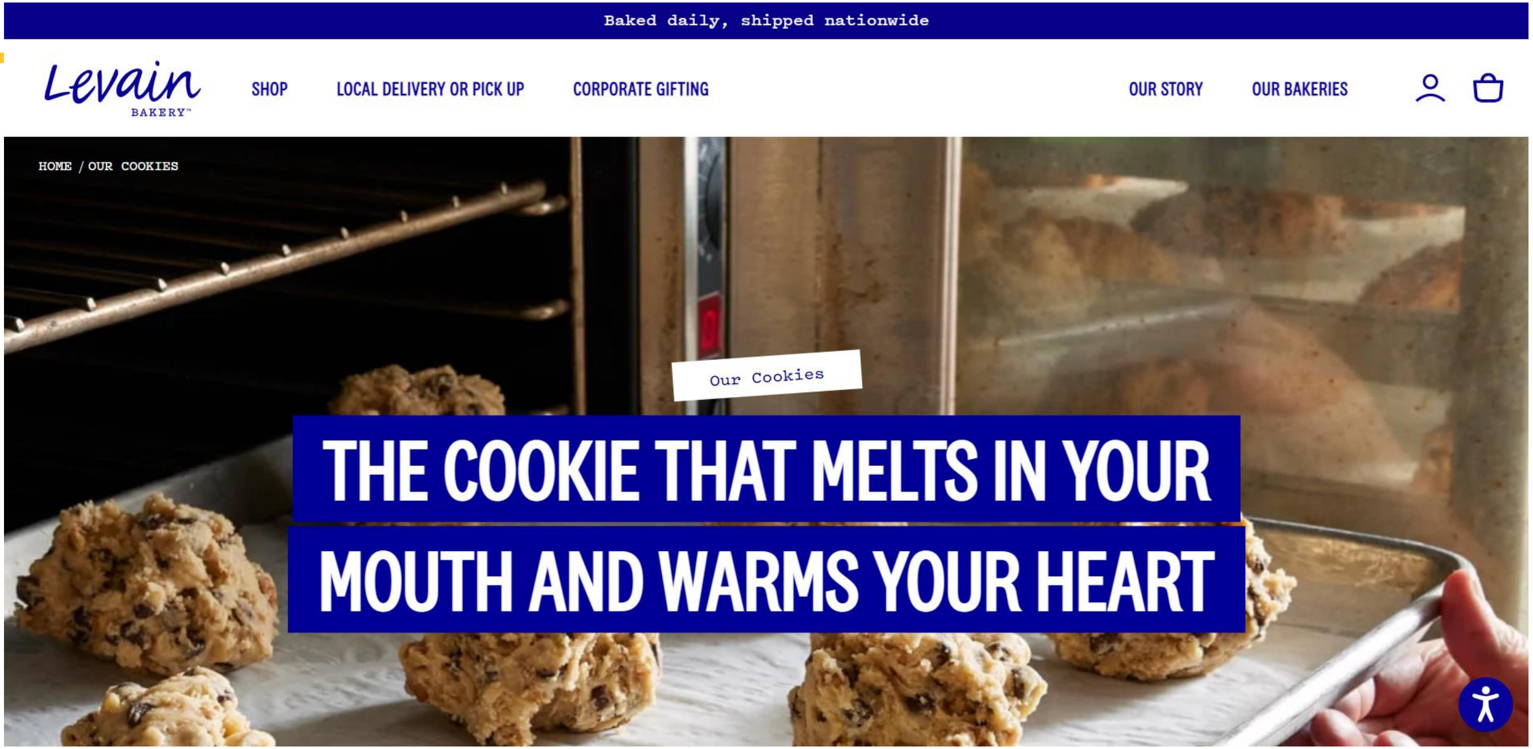
Do you have enough time to build it? What are the options? What do you compromise/remove?

Who, What, Why, When, How

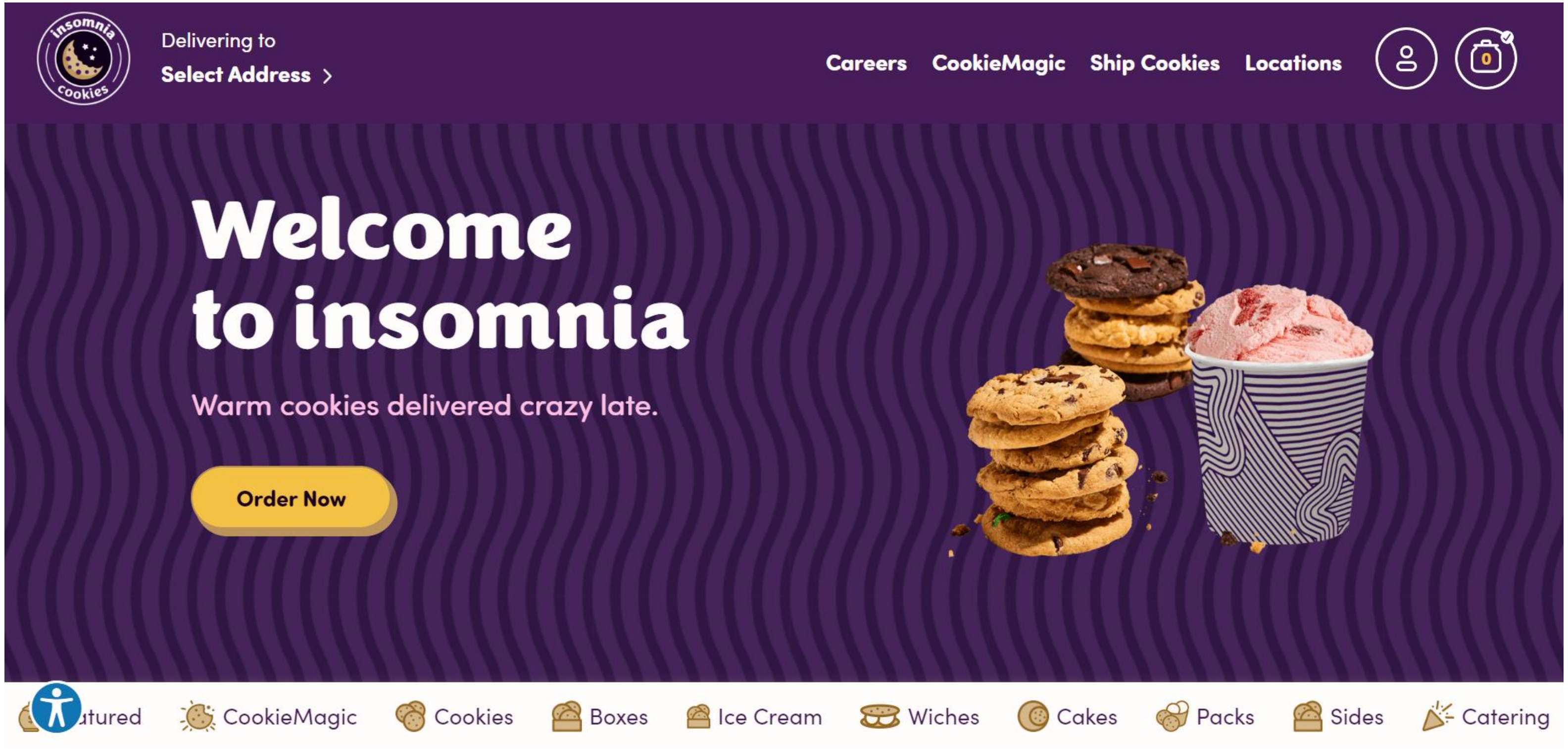
- What are you building
- Who will be using it
- Why are they using it
- How will they use it
- Where will they use it

- Why are you building it
- How do you build it
- Who will build it
- Where will you build it

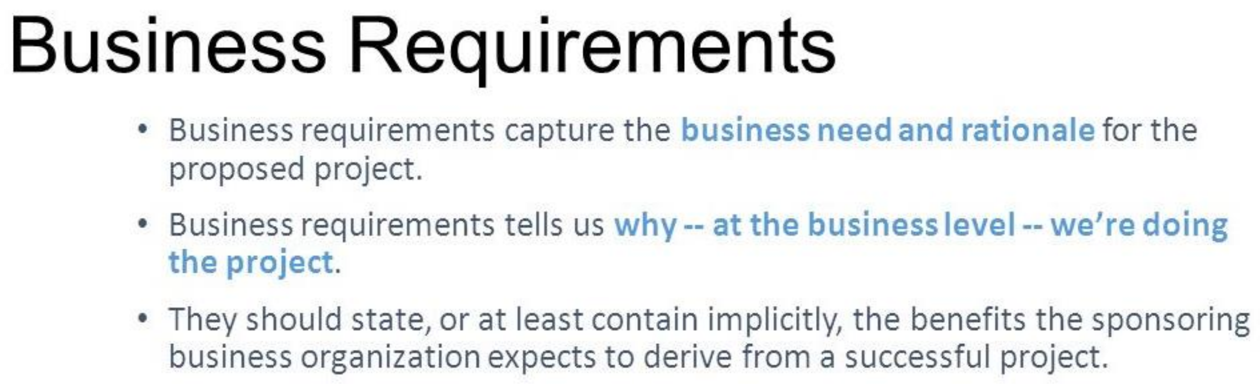
Business Objective


Baked daily, shipped nationwide
Levain BAKERY™
SHOP
LOCAL DELIVERY OR PICK UP
CORPORATE GIFTING
OUR STORY
OUR BAKERIES
HOME / OUR COOKIES
Our Cookies
THE COOKIE THAT MELTS IN YOUR MOUTH AND WARMS YOUR HEART

Business Objective



Business Requirements Gathering



Business Requirements

- Business requirements capture the **business need and rationale** for the proposed project.
- Business requirements tells us **why -- at the business level -- we're doing the project**.
- They should state, or at least contain implicitly, the benefits the sponsoring business organization expects to derive from a successful project.

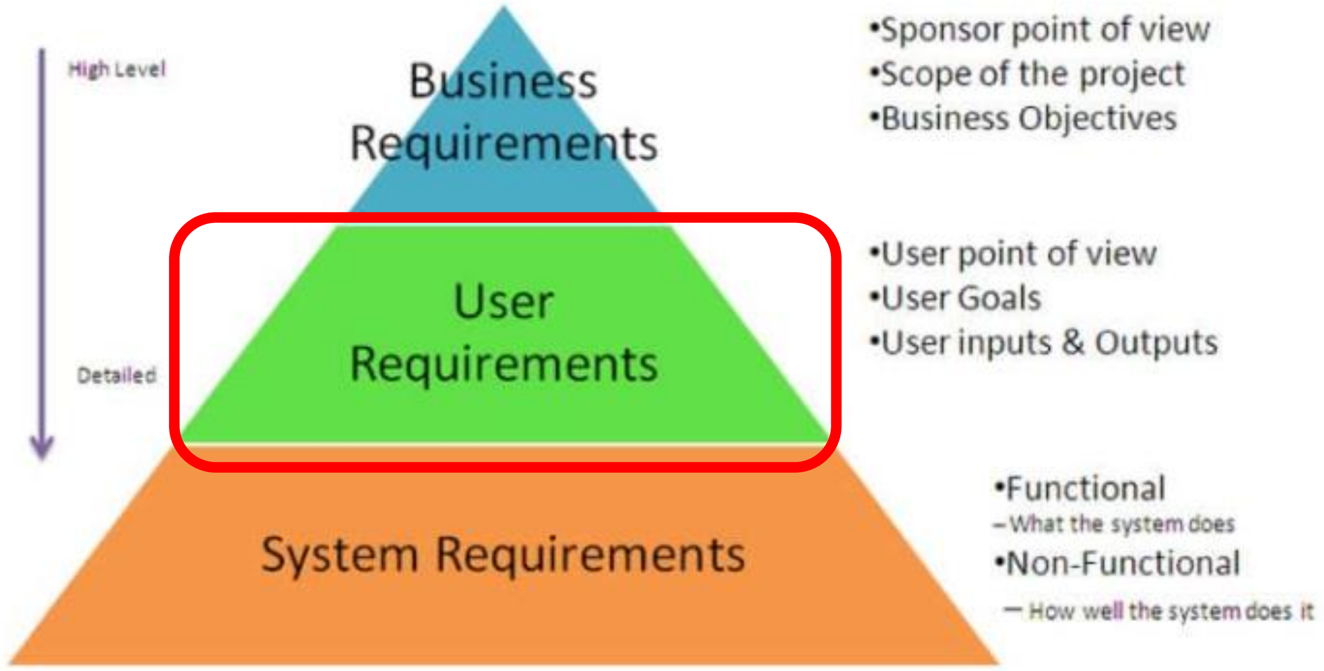
Business Requirements Gathering

Business View
Scope
Business Objectives

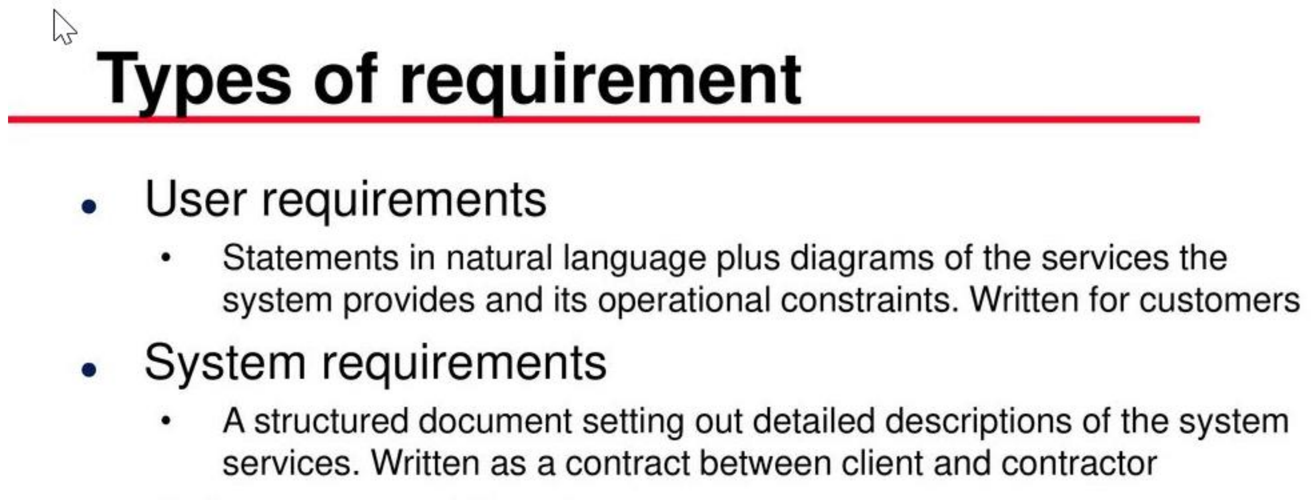
The new Cookie Company website will better serve our underserved cookie lovers and allowing them to select and order cookies of their choice for quick home or office delivery. Cookie lovers will get to custom order their selections and receive fresh cookies from the comfort of their homes or offices.

UNISYS | Securing Your Tomorrow®

Types of Requirements Gathering



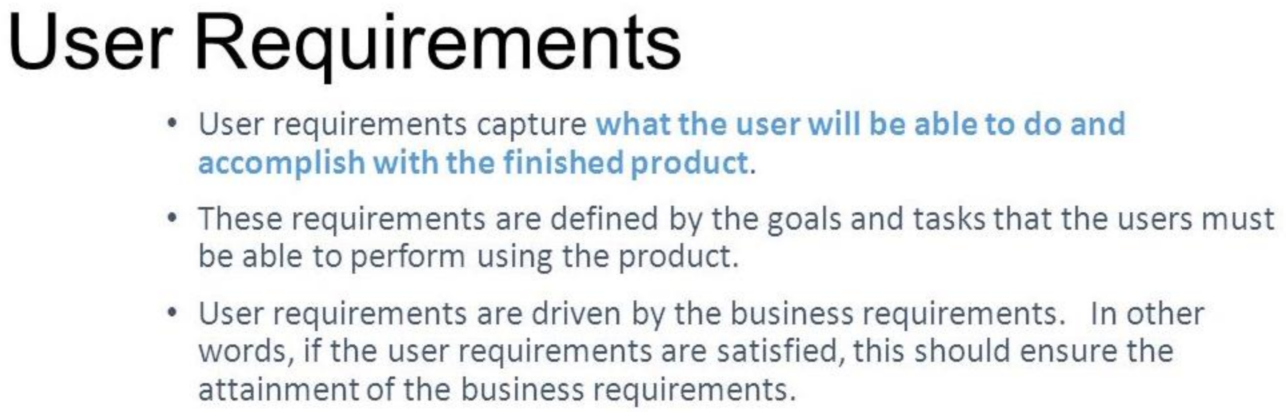
User & System Requirements Gathering



Types of requirement

- User requirements
  - Statements in natural language plus diagrams of the services the system provides and its operational constraints. Written for customers
- System requirements
  - A structured document setting out detailed descriptions of the system services. Written as a contract between client and contractor

User Requirements Gathering



User Requirements

- User requirements capture **what the user will be able to do and accomplish with the finished product.**
- These requirements are defined by the goals and tasks that the users must be able to perform using the product.
- User requirements are driven by the business requirements. In other words, if the user requirements are satisfied, this should ensure the attainment of the business requirements.

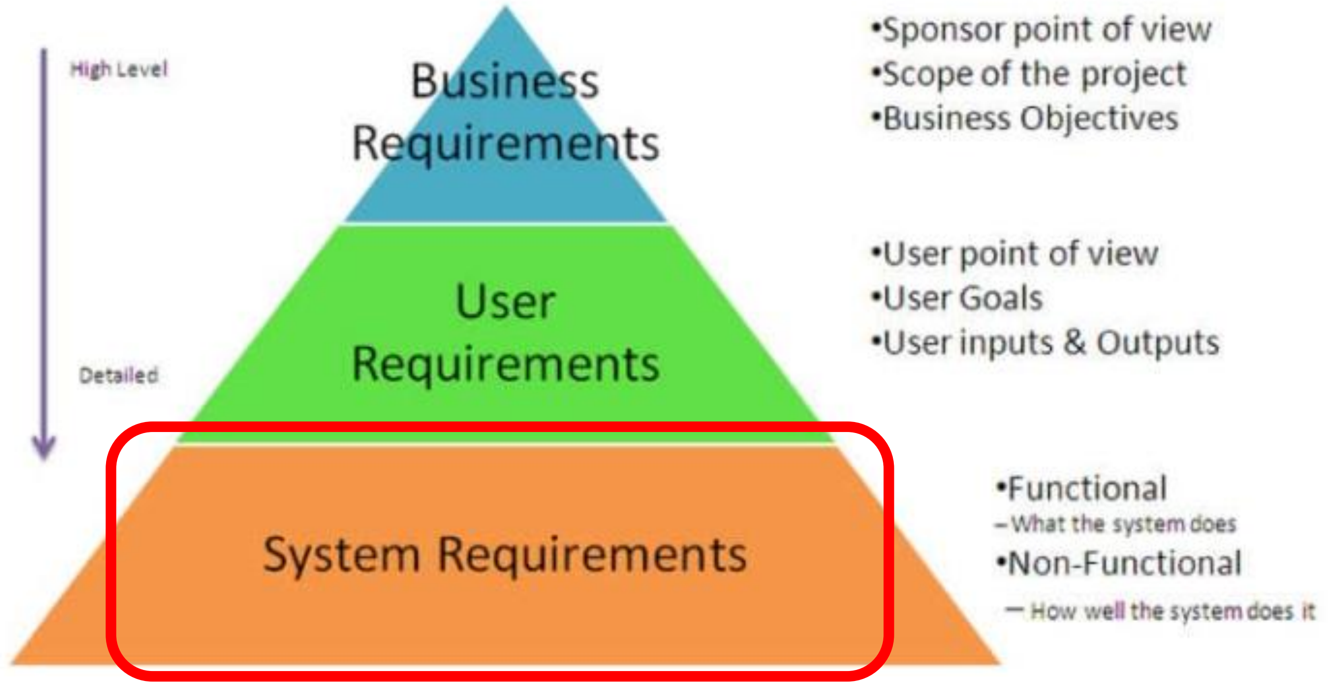
User Requirements Gathering

Users Point of View
User Goals
User input and output

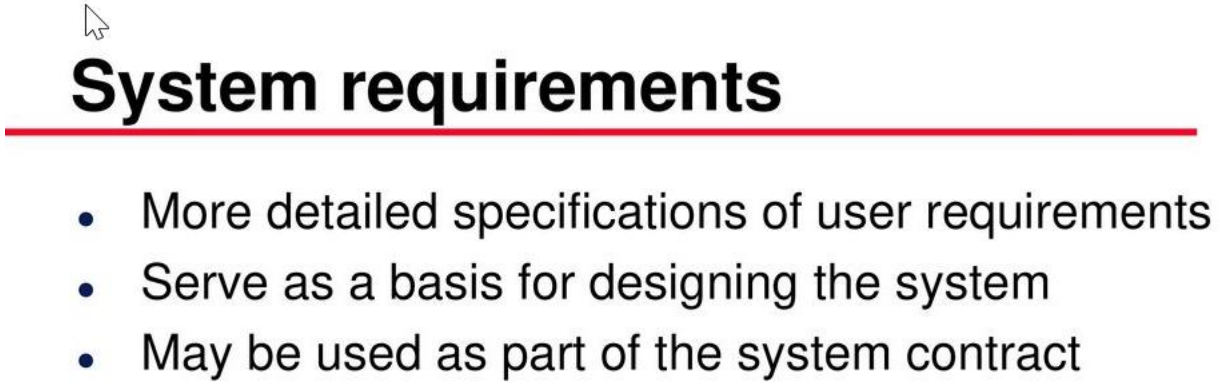
Customers will be able to:

- create accounts,
- view currently available cookie flavors,
- order cookies
- manage their orders,
- receive status of their orders
- Receive status of shipping information,
- online receipts,
- receive notices/specials

Types of Requirements Gathering



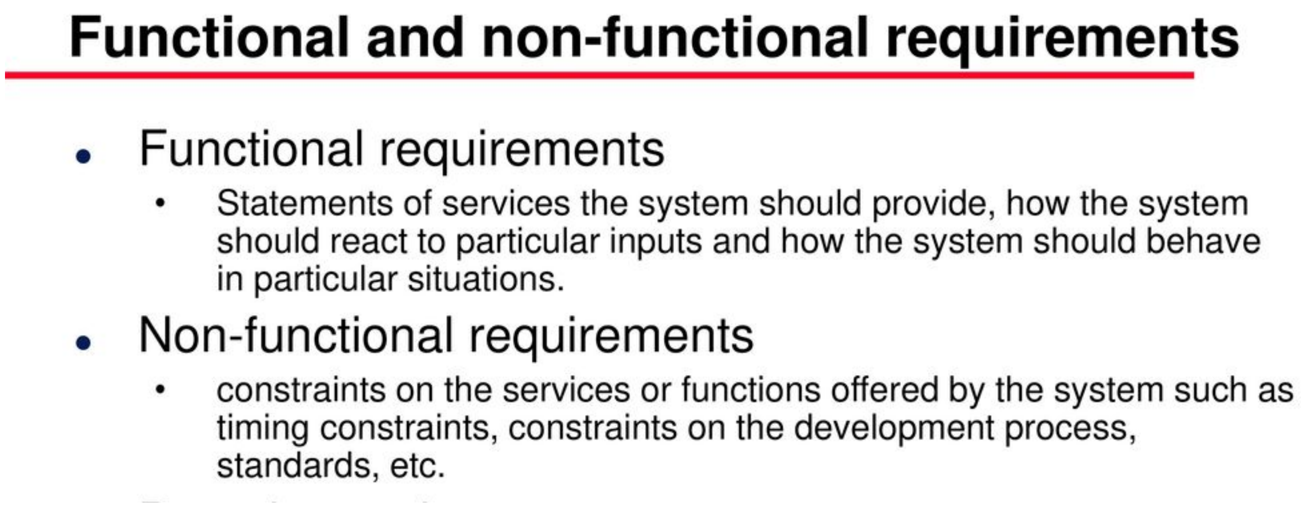
System Requirements Gathering



System requirements

- More detailed specifications of user requirements
- Serve as a basis for designing the system
- May be used as part of the system contract

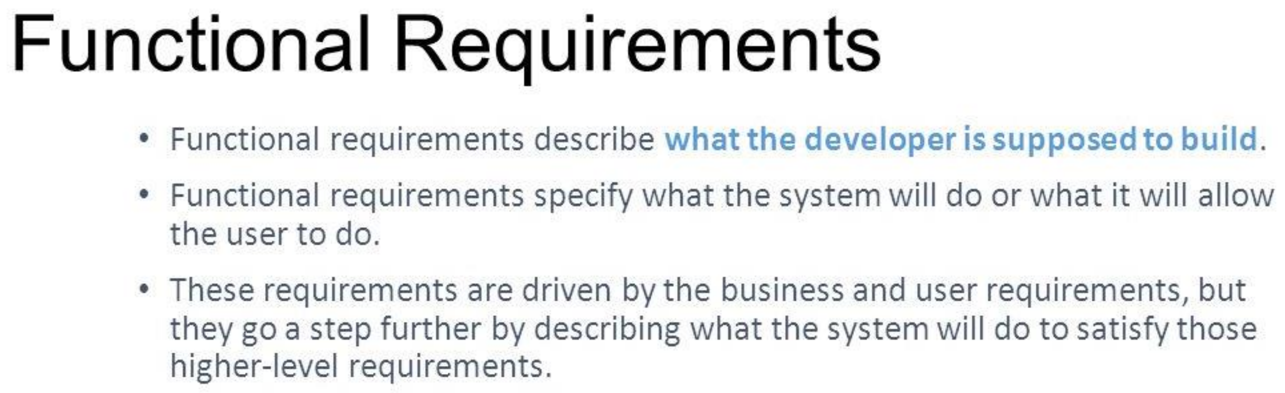
System Requirements Gathering



Functional and non-functional requirements

- Functional requirements
  - Statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.
- Non-functional requirements
  - constraints on the services or functions offered by the system such as timing constraints, constraints on the development process, standards, etc.

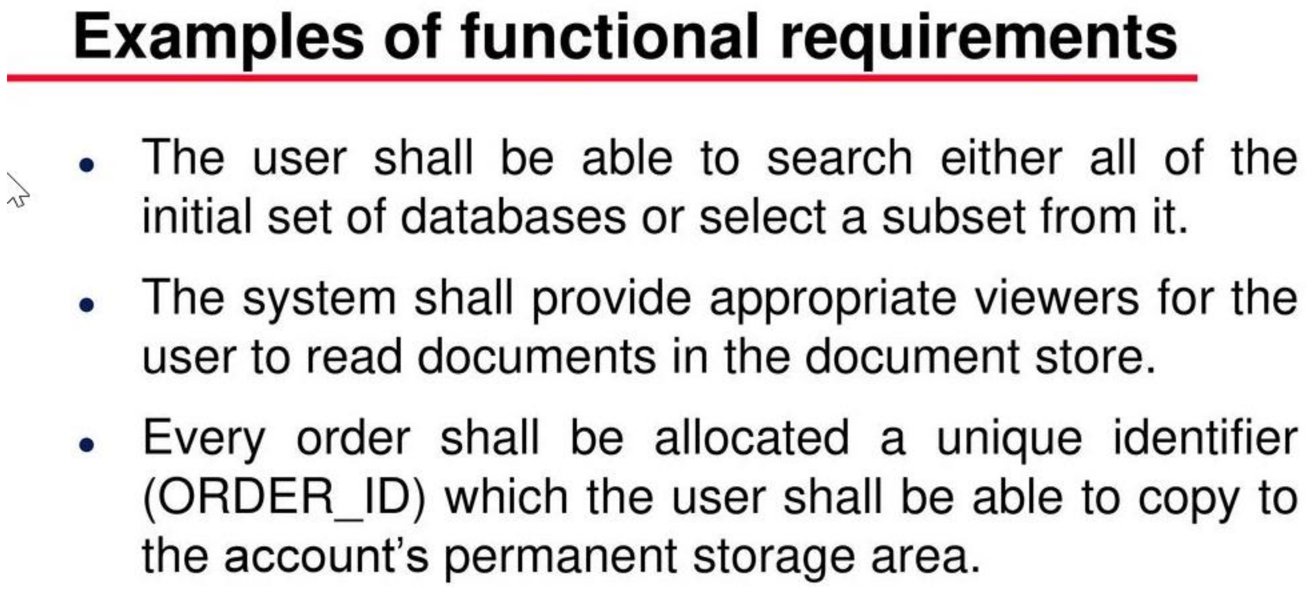
Functional Requirements Gathering



Functional Requirements

- Functional requirements describe **what the developer is supposed to build**.
- Functional requirements specify what the system will do or what it will allow the user to do.
- These requirements are driven by the business and user requirements, but they go a step further by describing what the system will do to satisfy those higher-level requirements.

Functional Requirements Gathering

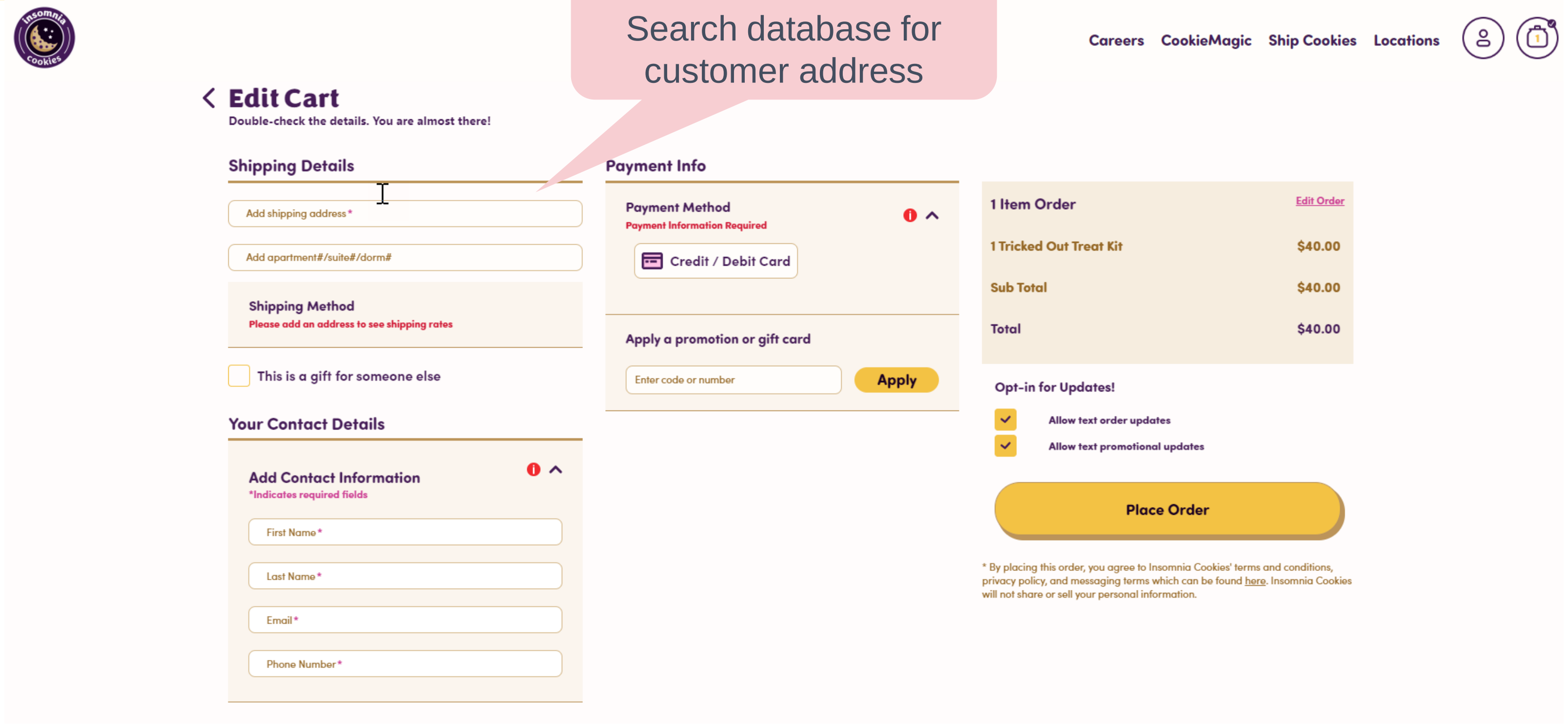


Examples of functional requirements

- The user shall be able to search either all of the initial set of databases or select a subset from it.
- The system shall provide appropriate viewers for the user to read documents in the document store.
- Every order shall be allocated a unique identifier (ORDER_ID) which the user shall be able to copy to the account's permanent storage area.

Our Website Example – Order Cookies

Function:
Search database for customer address


Careers CookieMagic Ship Cookies Locations

Edit Cart
Double-check the details. You are almost there!

Shipping Details

Add shipping address *

Add apartment#/suite#/dorm#

Shipping Method
Please add an address to see shipping rates

This is a gift for someone else

Your Contact Details

Add Contact Information
*Indicates required fields

First Name *

Last Name *

Email *

Phone Number *

Payment Info

Payment Method
Payment Information Required

Credit / Debit Card

Apply a promotion or gift card

Enter code or number

Apply

1 Item Order	Edit Order
1 Tricked Out Treat Kit	$40.00
Sub Total	$40.00
Total	$40.00

Opt-in for Updates!

✓ Allow text order updates

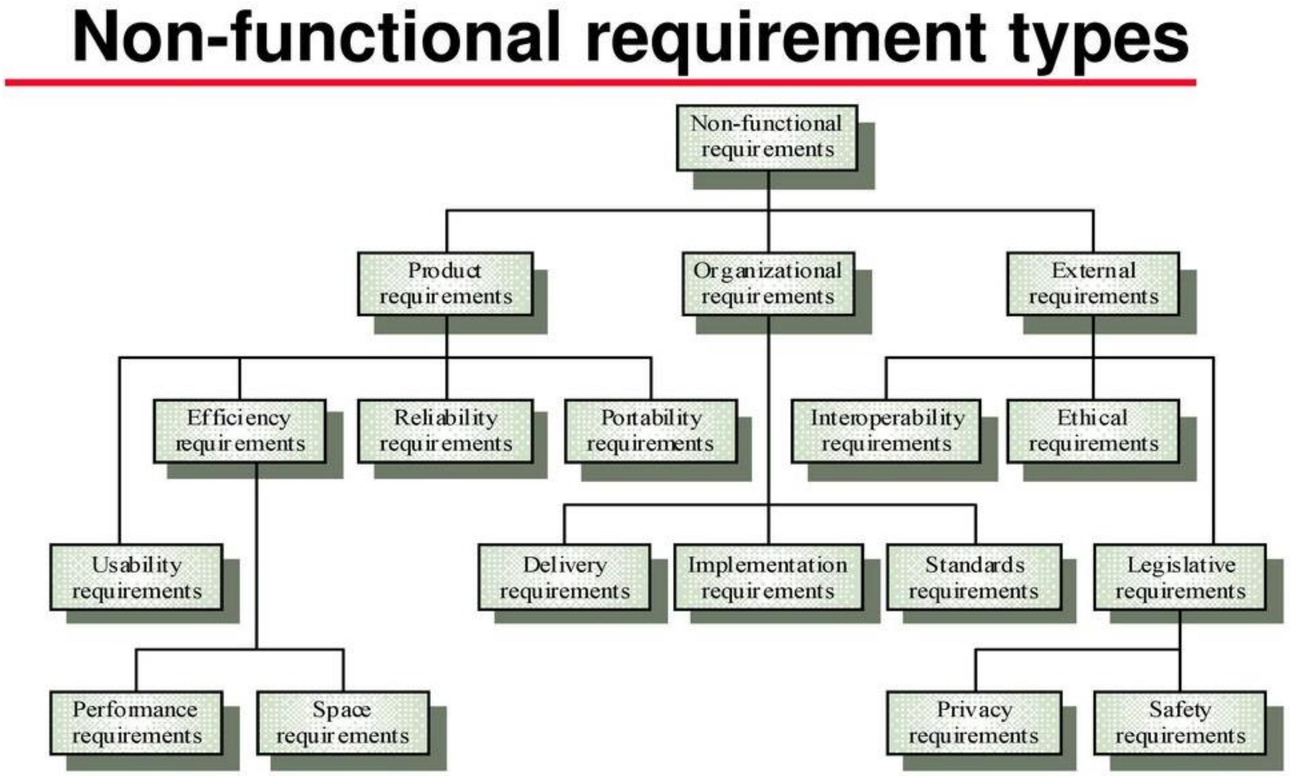
✓ Allow text promotional updates

Place Order

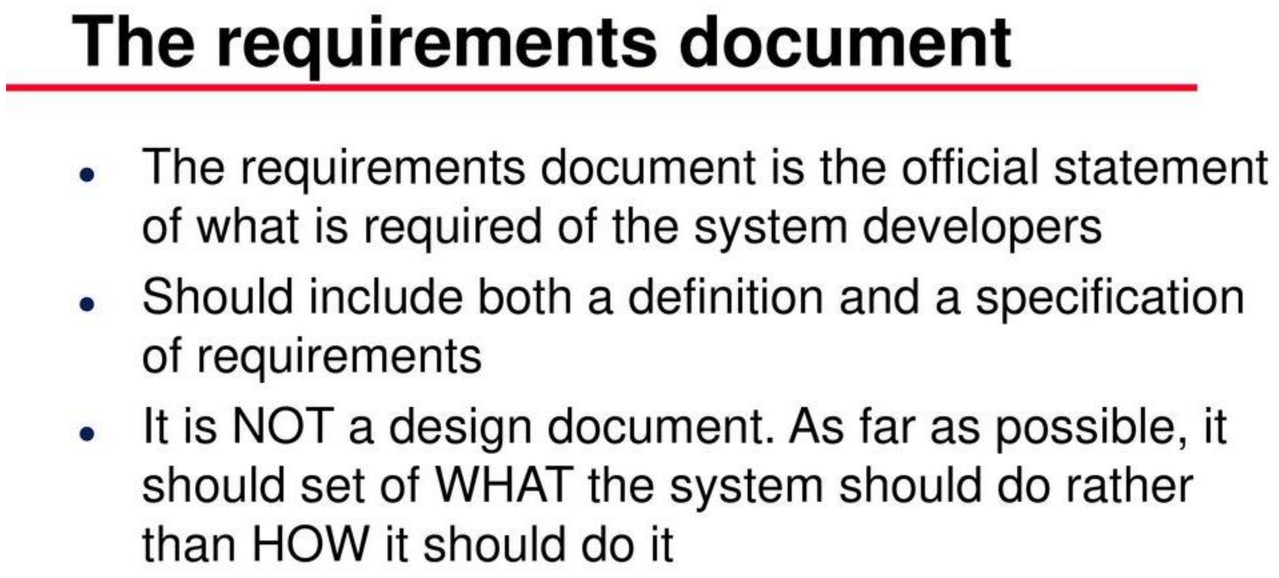
* By placing this order, you agree to Insomnia Cookies' terms and conditions, privacy policy, and messaging terms which can be found here. Insomnia Cookies will not share or sell your personal information.

UNISYS | Securing Your Tomorrow®

Non-Functional Requirements Gathering



The Requirements Document



The requirements document

- The requirements document is the official statement of what is required of the system developers
- Should include both a definition and a specification of requirements
- It is NOT a design document. As far as possible, it should set of WHAT the system should do rather than HOW it should do it

User Requirements Gathering

	Business Requirements Document (BRD)	Software Requirement Specifications (SRS)	Functional Requirement Specifications (FRS)
Other names		Product Requirements Document (PRD) and System Requirements Specification	Functional Specifications Document (FSD), Product Specification Document (PSD), Functional Specs (FS)
Created By	Business Analyst	Business/System Analyst	Business/System Analyst/Implementation Leads
Contains	High level business requirements and stakeholder requirements	Detailed functional requirements, non-functional requirements and use cases	Granular functional requirements, data flow and UML diagrams
Used By	Upper and middle management	Project managers, SMEs (subject matter experts), technical and implementation lead	Technical leads, development teams and testing teams.
Prepared in	Initiation phase	Planning phase	Planning phase
Answers	'Why' the requirements are being undertaken	'What' requirements must be fulfilled to satisfy business needs	'How' exactly the system is expected to function
Example	Improve efficiency by tracking the employee time in office	Proposed software will contain following modules: Login, Administrator, Employee and Reporting	Login module will contain fields like: Enter username, Enter password, Submit button

The Requirements Document

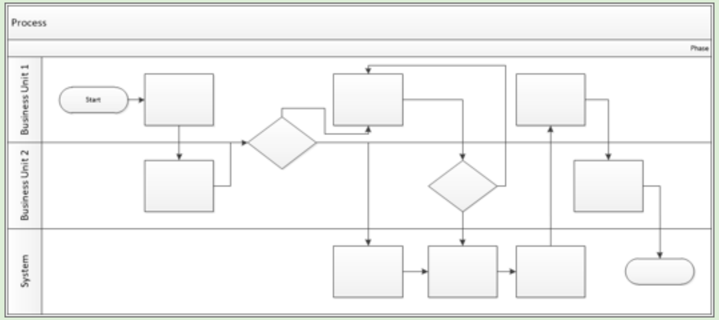
Introduction

- Project Summary
  - Project Objective
  - Background Info
  - Business Drivers
- Project Scope
  - In Scope/Out of Scope
    Functionality
- System Factors
  - Assumptions
  - Constraints
  - Risks
  - Issues

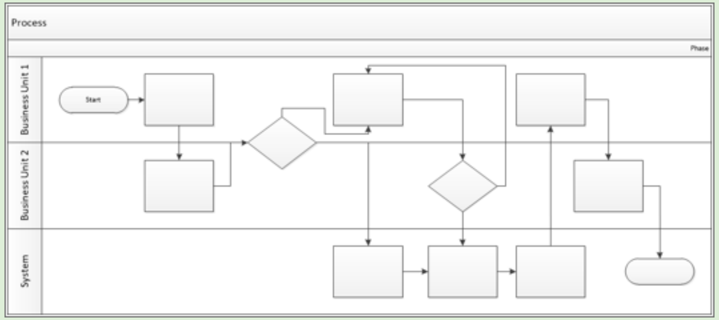
The Requirements Document

Business Processes

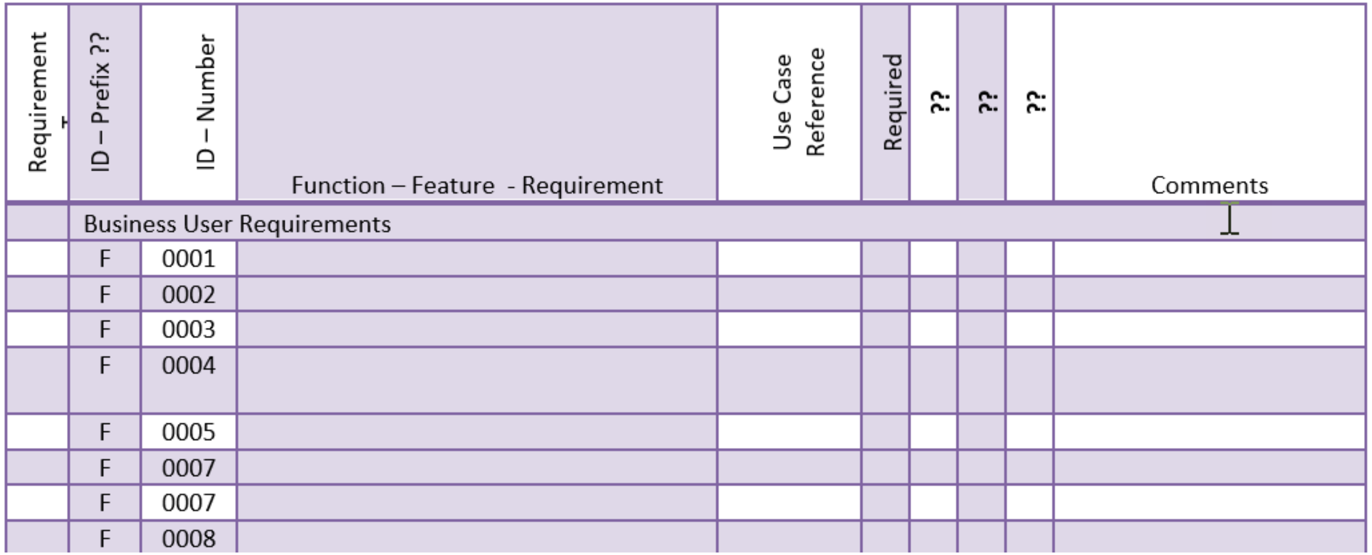
- Current Business Process (Now)



- Business Process (To-be)

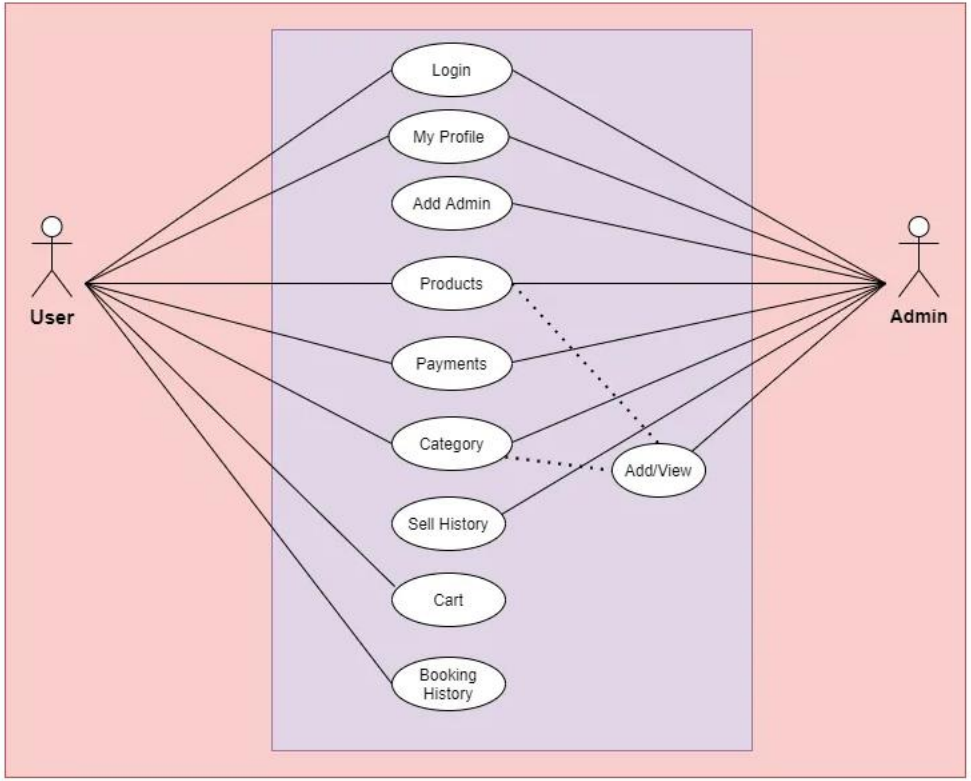


Business Requirements

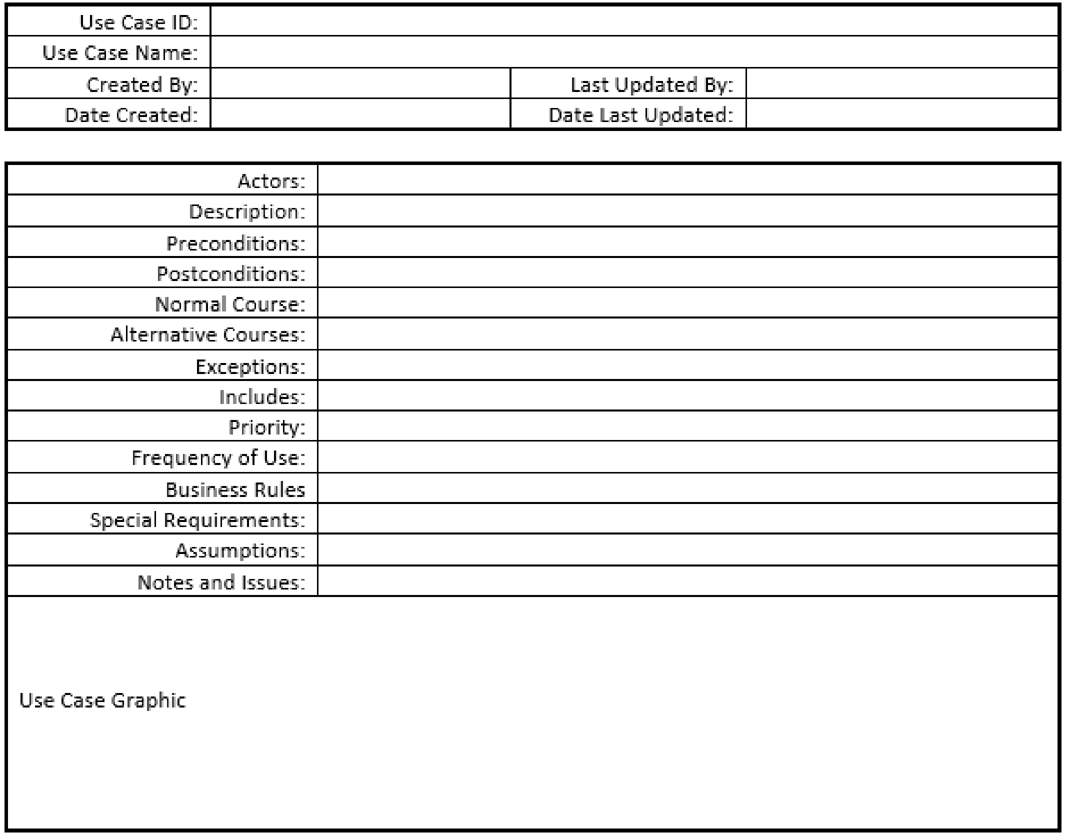


Requirement	ID – Prefix ??	ID – Number	Function – Feature - Requirement	Use Case Reference	Required	??	??	??	Comments
	Business User Requirements								
	F	0001							
	F	0002							
	F	0003							
	F	0004							
	F	0005							
	F	0007							
	F	0007							
	F	0008							

Use Case


User
Login
My Profile
Add Admin
Products
Payments
Category
Sell History
Cart
Booking History
Add/View
Admin

Use Cases



Use Case ID:			
Use Case Name:			
Created By:		Last Updated By:	
Date Created:		Date Last Updated:	

Actors:	
Description:	
Preconditions:	
Postconditions:	
Normal Course:	
Alternative Courses:	
Exceptions:	
Includes:	
Priority:	
Frequency of Use:	
Business Rules	
Special Requirements:	
Assumptions:	
Notes and Issues:	
Use Case Graphic	

|

System Specifications

Details, Details, Details

System Design

Overall Design Considerations

- Domain name
- Branding
- Functionality and Navigation
- Search Engine Optimization
- Responsive design
- Sitemap
  - Workflows
  - Pages
- Color palette
- Style guide

System Technical Design

Overall Operational/Technical Considerations

- Website hosting
- Web hosting tool
- Content Management
- Functionality and Navigation
- Search Engine Optimization
- Browser compatibility
- Responsive design
- Social media integration
- Security/Captcha
- Tracking/Metrics
- Sitemap

Data Design

Overall Data/Field Considerations

- Sources of data
- Source data types
- Sort order
- Database
- Tables
- Key
- Field types

Data Encoding

EBCDIC vs ASCII vs Unicode data encoding methods

- Extended Binary Coded Decimal Interchange Code (EBCDIC) is an 8-bit character encoding method for IBM mainframe machines.
- American Standard Code for Information Interchange (ASCII) is a 7-bit character encoding method for most other machines, including Windows, UNIX, and Macintosh machines
- The Unicode Standard **provides a unique number for every character, no matter what platform, device, application or language**. It has been adopted by all modern software providers and now allows data to be transported through many different platforms, devices and applications without corruption

Data Encoding

EBCDIC vs ASCII vs Unicode

The major difference is the numbers (or digits) are sorted after the alphabet (with lower case being sorted before upper case) for the EBCDIC environment.

The numbers (or digits) are sorted before the alphabet (with upper case being sorted before lower case) for the ASCII environment.

EBCDIC	ASCII
Special Characters	Special Characters
Lower Case	Numbers
Upper Case	Upper Case
Numbers	Lower Case

Data Design

Overall Data/Field Considerations

Field Types

- Text – 1 byte – 0 to 255
- Numeric
  - Integer – 2 bytes - -32,768 to 32,767
  - Long – 4 bytes - -2,147,483,648 to 2, 147,483,647
  - Floating point – 4 or 8 bytes - decimal
- General/Memo – up to 65,536 characters
- Currency – 8 bytes – 15 digit whole dollar plus 4 decimal places
- Date – 8 bytes – date and time
- Yes/No – 1 bit - binary

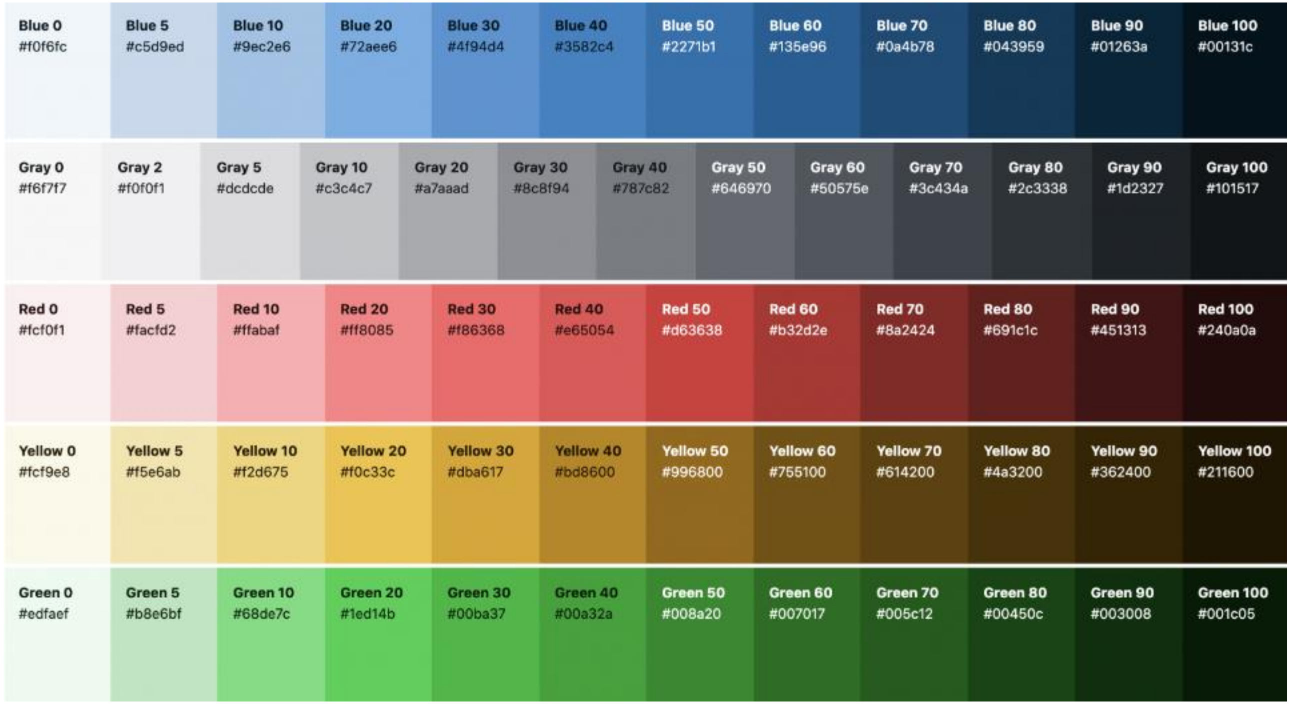
Table Design

Overall Table Considerations

Table

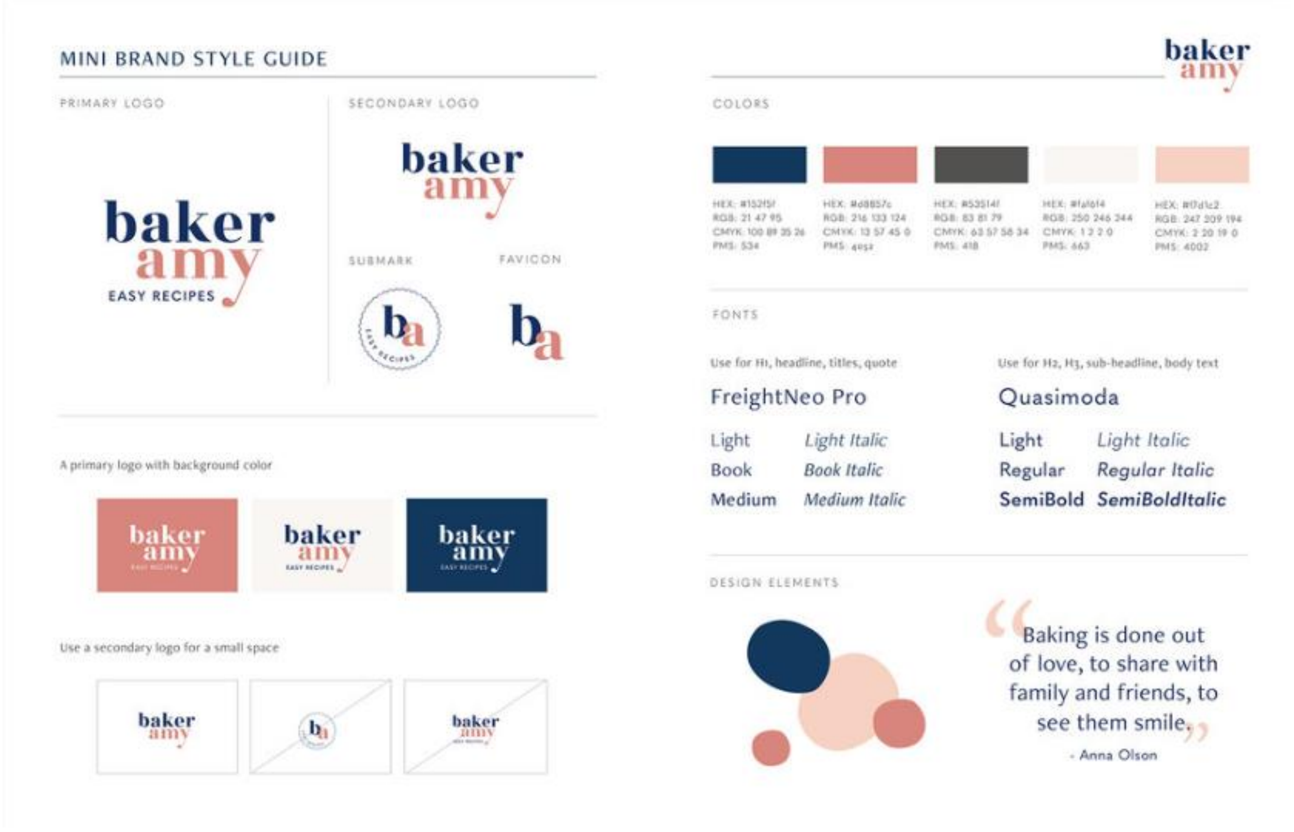
- Fields
- Keys
- Sort order

Color Palette



Blue 0 #f0f6fc	Blue 5 #c5d9ed	Blue 10 #9ec2e6	Blue 20 #72aee6	Blue 30 #4f94d4	Blue 40 #3582c4	Blue 50 #2271b1	Blue 60 #135e96	Blue 70 #0a4b78	Blue 80 #043959	Blue 90 #01263a	Blue 100 #00131c	
Gray 0 #f6f7f7	Gray 2 #f0f0f1	Gray 5 #dcdcde	Gray 10 #c3c4c7	Gray 20 #a7aaad	Gray 30 #8c8f94	Gray 40 #787c82	Gray 50 #646970	Gray 60 #50575e	Gray 70 #3c434a	Gray 80 #2c3338	Gray 90 #1d2327	Gray 100 #101517
Red 0 #fcf0f1	Red 5 #facfd2	Red 10 #ffabaf	Red 20 #ff8085	Red 30 #f86368	Red 40 #e65054	Red 50 #d63638	Red 60 #b32d2e	Red 70 #8a2424	Red 80 #691c1c	Red 90 #451313	Red 100 #240a0a	
Yellow 0 #fcf9e8	Yellow 5 #f5e6ab	Yellow 10 #f2d675	Yellow 20 #f0c33c	Yellow 30 #dba617	Yellow 40 #bd8600	Yellow 50 #996800	Yellow 60 #755100	Yellow 70 #614200	Yellow 80 #4a3200	Yellow 90 #362400	Yellow 100 #211600	
Green 0 #edfaef	Green 5 #b8e6bf	Green 10 #68de7c	Green 20 #1ed14b	Green 30 #00ba37	Green 40 #00a32a	Green 50 #008a20	Green 60 #007017	Green 70 #005c12	Green 80 #00450c	Green 90 #003008	Green 100 #001c05	

Style Guide



MINI BRAND STYLE GUIDE

PRIMARY LOGO

baker amy
EASY RECIPES

SECONDARY LOGO

baker amy

SUBMARK

ba
EASY RECIPES

FAVICON

ba

A primary logo with background color

baker amy EASY RECIPES | baker amy EASY RECIPES | baker amy EASY RECIPES

Use a secondary logo for a small space

baker amy | ba | baker amy

baker amy

COLORS

🟦	🟥	⬛	⬜	🟧
HEX: #152f5f	HEX: #d8857c	HEX: #53514f	HEX: #faf6f4	HEX: #f7d1c2
RGB: 21 47 95	RGB: 216 133 124	RGB: 83 81 79	RGB: 250 246 244	RGB: 247 209 194
CMYK: 100 89 35 26	CMYK: 13 57 45 0	CMYK: 63 57 58 34	CMYK: 1 2 2 0	CMYK: 2 20 19 0
PMS: 534	PMS: 4052	PMS: 418	PMS: 663	PMS: 4002

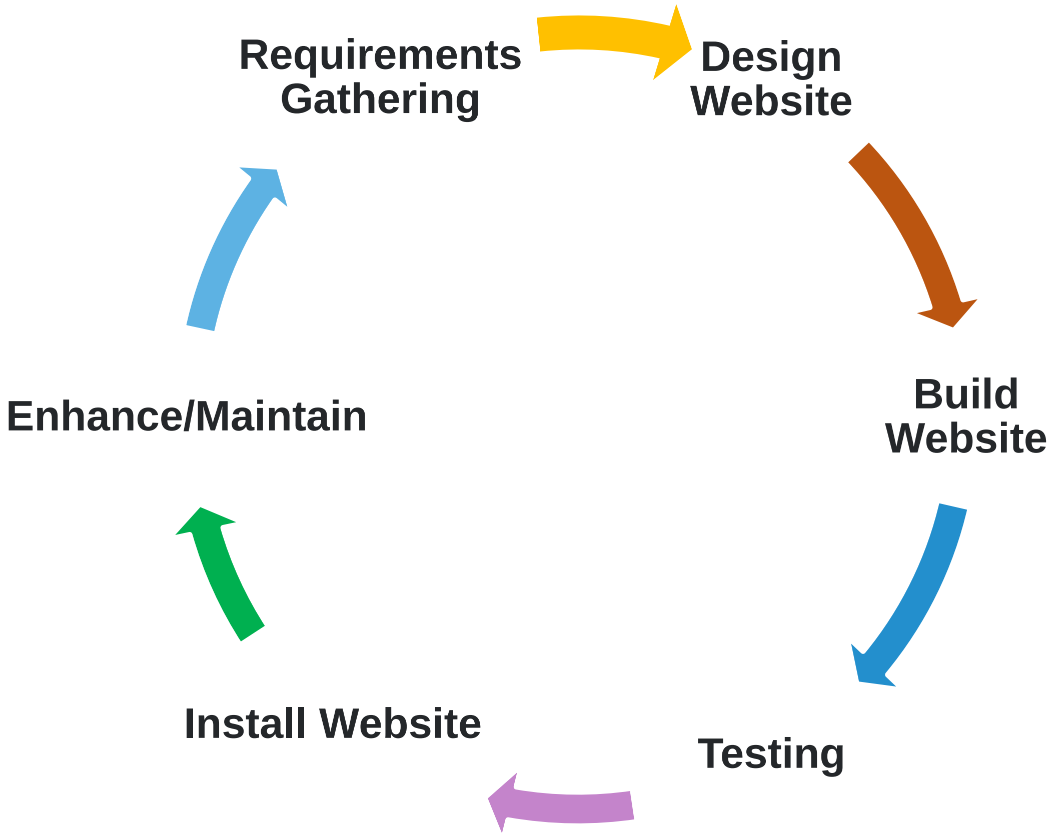
FONTS

Use for H1, headline, titles, quote		Use for H2, H3, sub-headline, body text	
FreightNeo Pro		Quasimoda	
Light	*Light Italic*	Light	*Light Italic*
Book	*Book Italic*	Regular	*Regular Italic*
Medium	*Medium Italic*	**SemiBold**	***SemiBoldItalic***

DESIGN ELEMENTS

“Baking is done out of love, to share with family and friends, to see them smile.”
- Anna Olson

Website Development Lifecycle

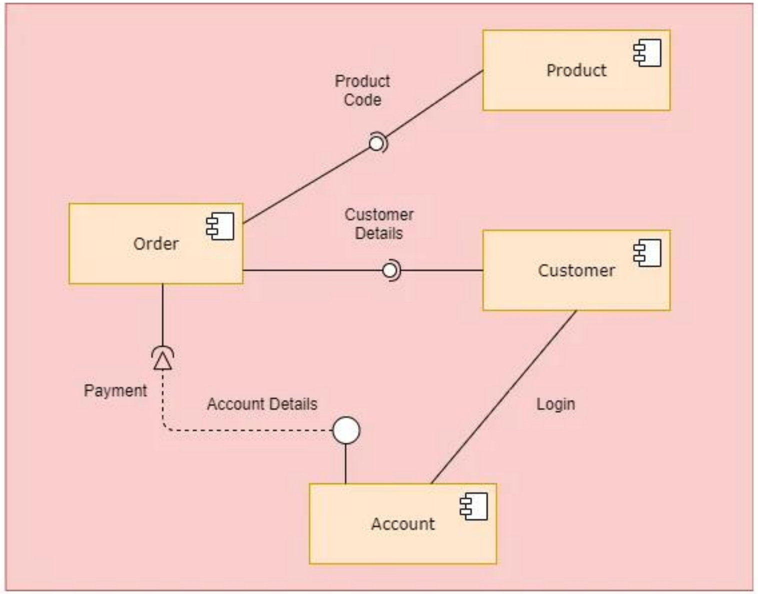


UNISYS | Securing Your Tomorrow®

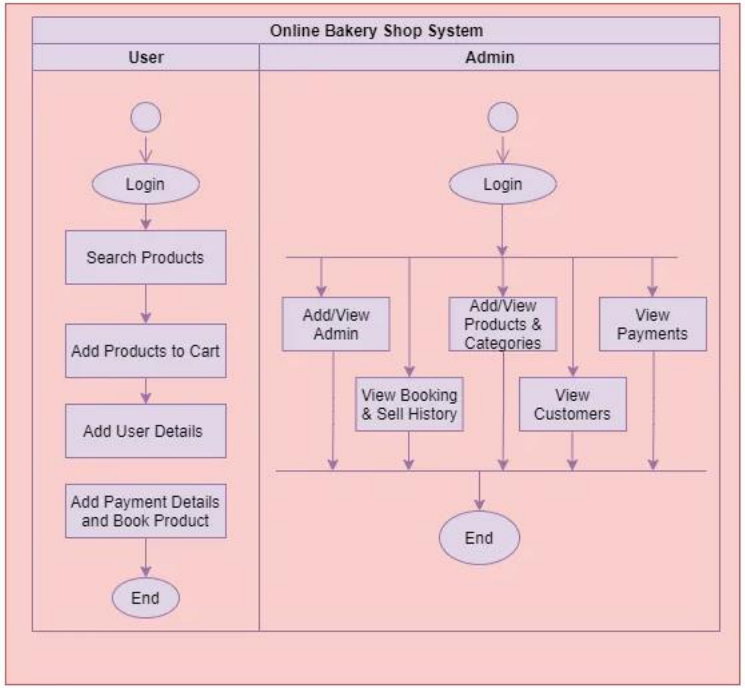
QUESTIONS?

Backup Slides

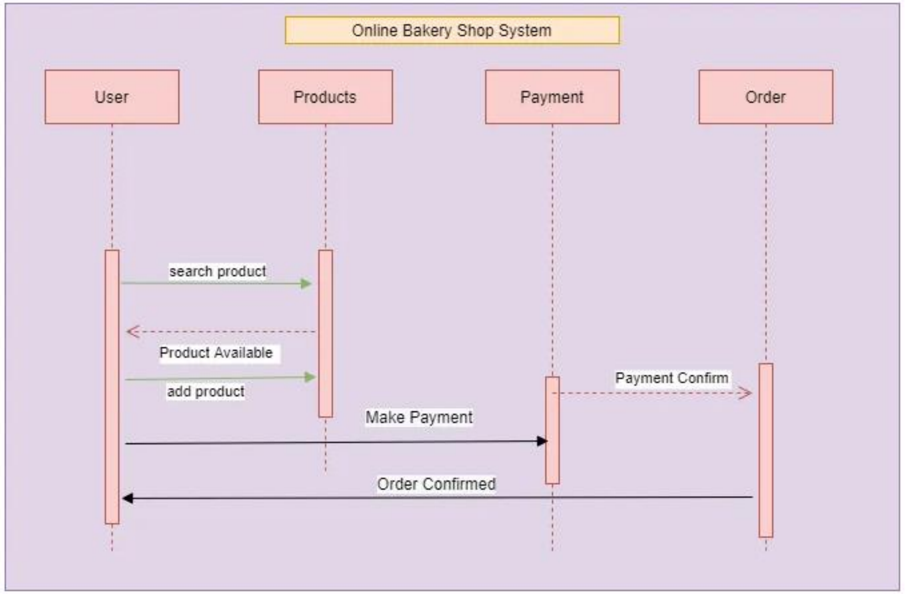
Component Diagram



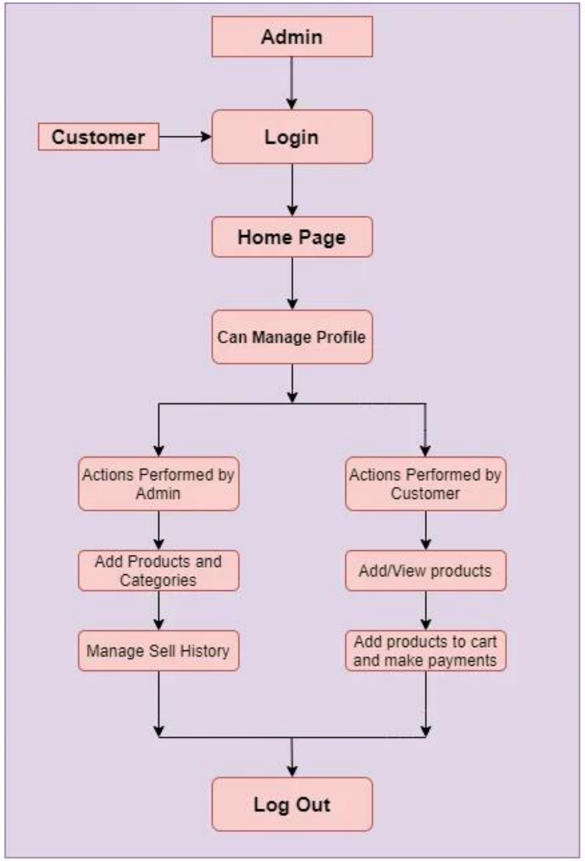
Workflow/Activity


Online Bakery Shop System
User
Admin
Login
Search Products
Add Products to Cart
Add User Details
Add Payment Details and Book Product
End
Login
Add/View Admin
View Booking & Sell History
Add/View Products & Categories
View Customers
View Payments
End

Sequence/Workflow



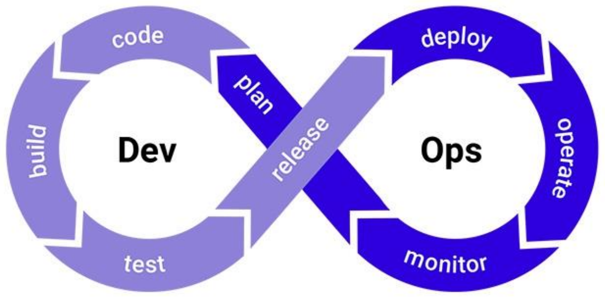
Data Flow Diagram


Admin
Customer
Login
Home Page
Can Manage Profile
Actions Performed by Admin
Actions Performed by Customer
Add Products and Categories
Add/View products
Manage Sell History
Add products to cart and make payments
Log Out

Software Development Lifecycle

DevOps deployment methodology

DevOps is not just a development methodology but also a set of practices that supports an organizational culture. DevOps deployment centers on organizational change that enhances collaboration between the departments responsible for different segments of the development life cycle, such as development, quality assurance, and operations.



Software Development Lifecycle

Pros: DevOps is focused on improving time to market, lowering the failure rate of new releases, shortening the lead time between fixes, and minimizing disruption while maximizing reliability. To achieve this, DevOps organizations aim to automate continuous deployment to ensure everything happens smoothly and reliably. Companies that use DevOps methods benefit by significantly reducing time to market and improving customer satisfaction, product quality, and employee productivity and efficiency.

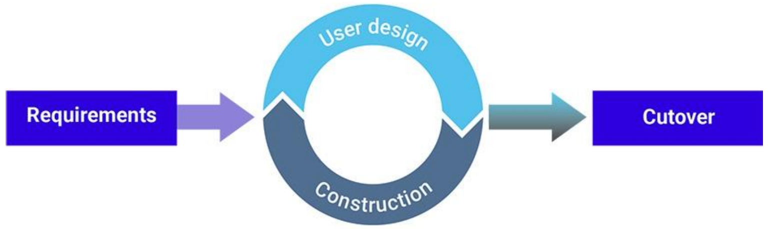
Cons: Even in light of its benefits, there are a few drawbacks to DevOps:
•Some customers don't want continuous updates to their systems.
•Some industries have regulations that require extensive testing before a project can move to the operations phase.
•If different departments use different environments, undetected issues can slip into production.
•Some quality attributes require human interaction, which slows down the delivery pipeline.

Software Development Lifecycle

Rapid application development

Rapid application development (RAD) is a condensed development process that produces a high-quality system with low investment costs. Scott Stiner, CEO and president of UM Technologies, said in Forbes, “This RAD process allows our developers to quickly adjust to shifting requirements in a fast-paced and constantly changing market.” The ability to quickly adjust is what allows such a low investment cost.

The rapid application development method contains four phases: requirements planning, user design, construction, and cutover. The user design and construction phases repeat until the user confirms that the product meets all requirements.



Software Development Lifecycle

Pros: Rapid application development is most effective for projects with a well-defined business objective and a clearly defined user group, but which are not computationally complex. RAD is especially useful for small to medium projects that are time sensitive.

Cons: Rapid application development requires a stable team composition with highly skilled developers and users who are deeply knowledgeable about the application area. Deep knowledge is essential in a condensed development timeline that requires approval after each construction phase. Organizations that don’t meet these requirements are unlikely to benefit from RAD.