Project Discovery

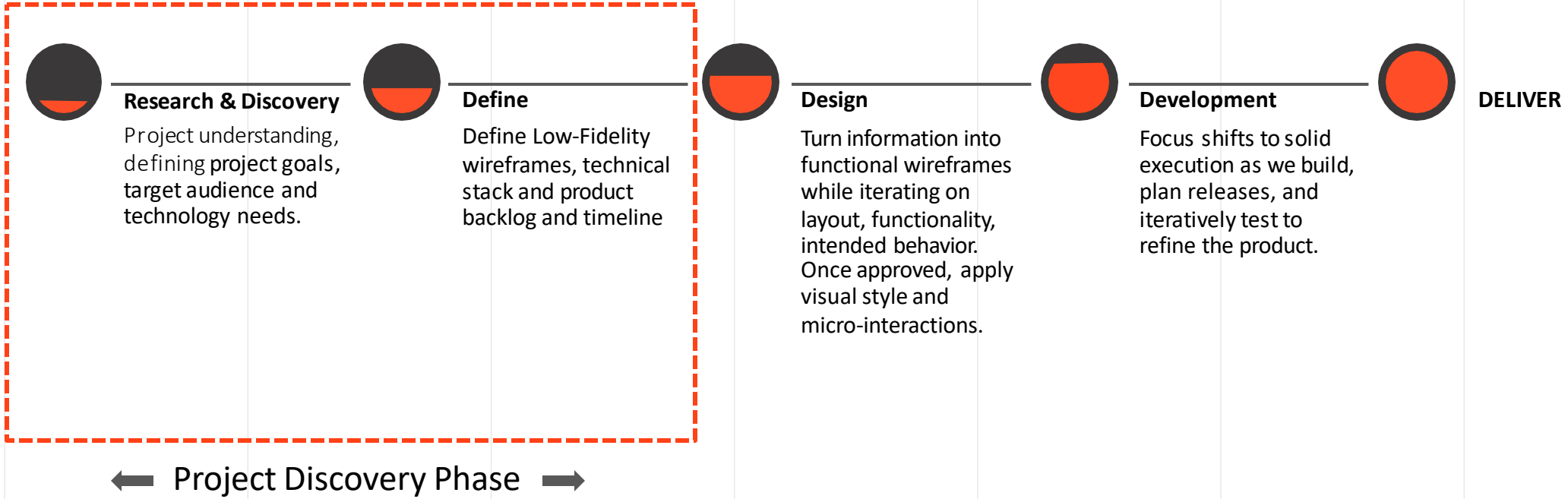
— Billing Platform

Client: Non-Disclosed

Date: June 2020

Phoenix Consultancy

Project Discovery - Explained



Phoenix Consultancy

Progress to Date

Project Domain	Project Artifact	Status	Link
Technology	Preliminary Tech Stack Definition	**Created**	
UX	Design Principles	**Created**	
UX	Personas	**Created**	
UX	User Journeys	**Created**	
UX	Preliminary Site Map	**Created**	
UX	Wireframes	**Created**	
PM	Epics and User Stories	**Created**	
PM	Project Plan (Timeline, Staffing, Estimates)	**Created**	

Phoenix Consultancy

Project Overview

Phoenix Consultancy

Summary

Project Overview

When operating in any IOT business, company, billing the customers becomes one the most important parts of the system. As it is usually charged by percentage of the revenue, it is considered expensive. In addition, most billing platforms are not flexible, making them unable to fulfill company's specific needs. Everything mentioned lead this company to switch course and create tailor-made billing software platform.

Primary function of the software is to invoice customers based on logs of the traffic used on their internal devices. This platform automates collecting and normalizing multiple data sources from various operators and while providing visually impactful and feature-rich dashboard, it enables users to view the data KPIs from various perspectives within the data hierarchy.

Unlike other billing platforms, fundamental to this one is the ability to quickly and easily configure the system to suit every aspect of the enterprise billing operation. The extensible architecture will help company streamline its operations today and scale for the future, thus becoming an important company asset with tendency to be marketed in the next phases of its life cycle.

Target Audience

- Companies operating in TelCo industry
- Companies' internal users:
    - Finance administrators
    - CFOs
    - General managers

Phoenix Consultancy

Goals

1. Reduce overall costs
2. Increase flexibility
3. Create bespoke functionalities
4. Reduce overcharging by real-time billing
5. Eliminate invoicing errors
6. Sell the software to other companies

Phoenix Consultancy

Software functionality objectives

Billing Prorating

By incorporating the prorating way of billing, changing the subscription plans becomes less complicated and more accurate.
Billing platform uses CDRs to calculate the exact amount customer owes based on the amount of traffic that had been used over a particular time period.

High-level Customization

Custom-created features allow setting prorating rules, creating and updating packages and bundles, setting different billing types, generating different types of reports and more.

Bundles Management

Bundle management implies system flexibility by giving the platform users ability to activate and deactivate a bundle, assign or remove packages from bundles, as well as to allow package add-ons to be included in the bundle.

Enterprise level of interface

Building complex system such as billing platform makes integrations with other systems inevitable. Systems that software will be integrated with must be chosen attentively, making sure they provide all needed aspects, as well as the expected level of security and performance.

Real Time Notifications System

Creating reliable and punctual notification system gives both customers and company relevant, just-in-time information.
For customers, it is the alert when the threshold is nearly reached or invoice notification. For company, the alerts are system-oriented, notifying relevant people about possible bundle or traffic issues, billing details or similar, allowing them to act proactively.

Phoenix Consultancy

Billing Concept

1. **Broadbands**

Broadband is the smallest unit/entity in billing concept. It is a SIM card with certain amount of data that will form a package.

2. **Packages**

Package is made of one or several broadbands merged in order to combine different benefits selected broadbands have. Each package can be upgraded with different add-ons or revised anytime.

3. **Bundles**

Bundle is the actual amount of data customer will receive. It is consisted of various packages, which allows additional flexibility in creating numerous bundle types, that will match customers' needs.

3. **Billing Types**

Customers are being billed differently, depending on the bundle they have chosen. Supported billing types are:

- Recurring
- One off

Phoenix Consultancy

Billing Concept Examples

Regular Bundle

Simpliest bundle type is Regular, containing one or more packages without any add on option. Once the bundle threshold is reached, customers will get charged for the overage in case they continue to use the data.

Customers are invoiced at the end of the accounting period (usually monthly) for the number of packages used for the bundle plus the overage, if any.

Bundle + Add On

This type of bundle is similar to the Regular Bundle, with the exception of customers being notified when they reach 80% of the threshold and given the option to top up their bundle with one of proposed flat add-ons.

Customers are invoiced for the number of packages used for the bundle plus the chosen add on.

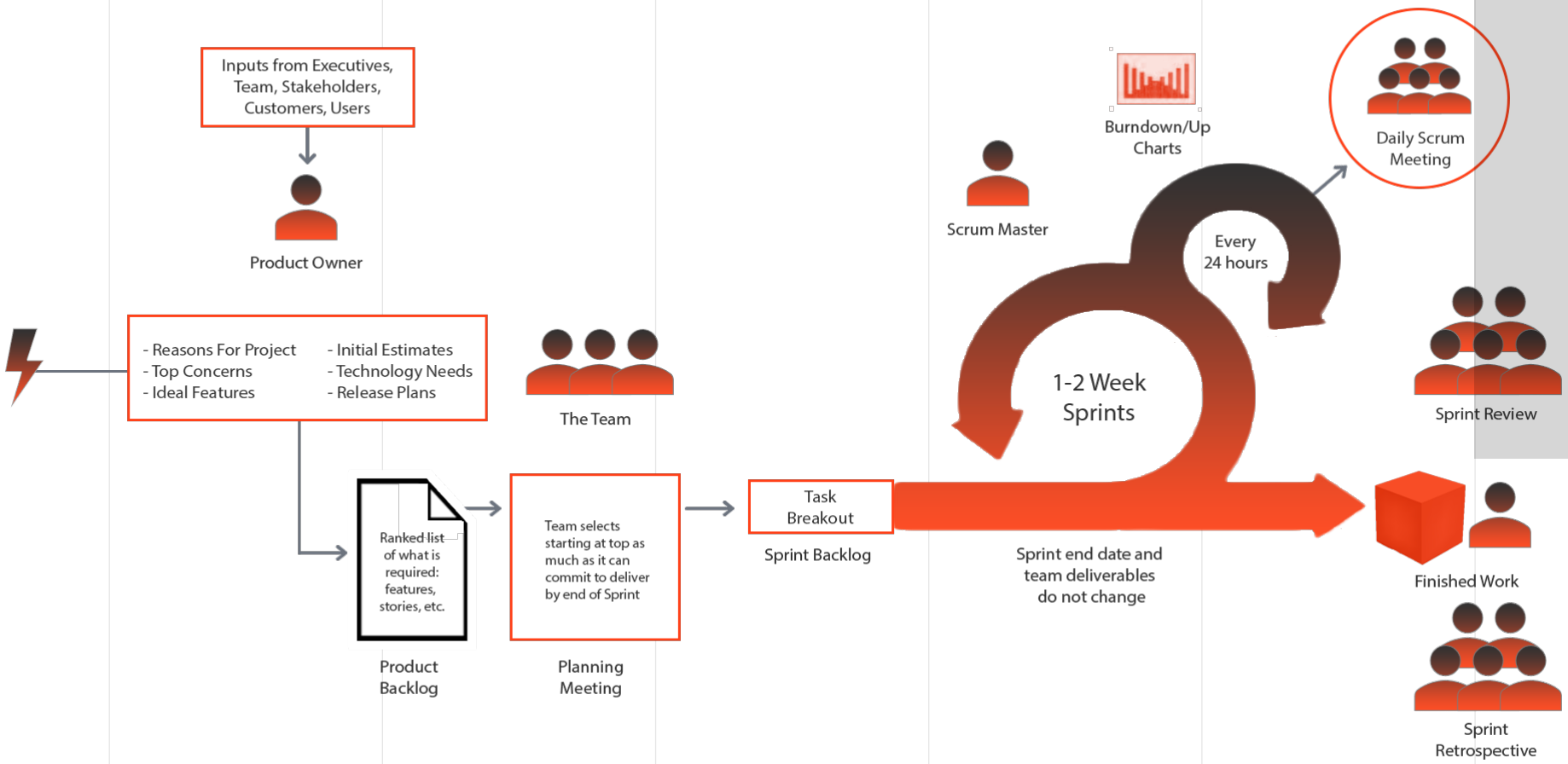
One Off Charges

Alongside with MRC (monthly recurring charges), defined by the bundle type, customers are periodically being imposed with one off charges. These charges include:

- Installation
- Activation
- Deactivation

Phoenix Consultancy

Agile Methodology Implementation



Phoenix Consultancy

Development Process

Discovery

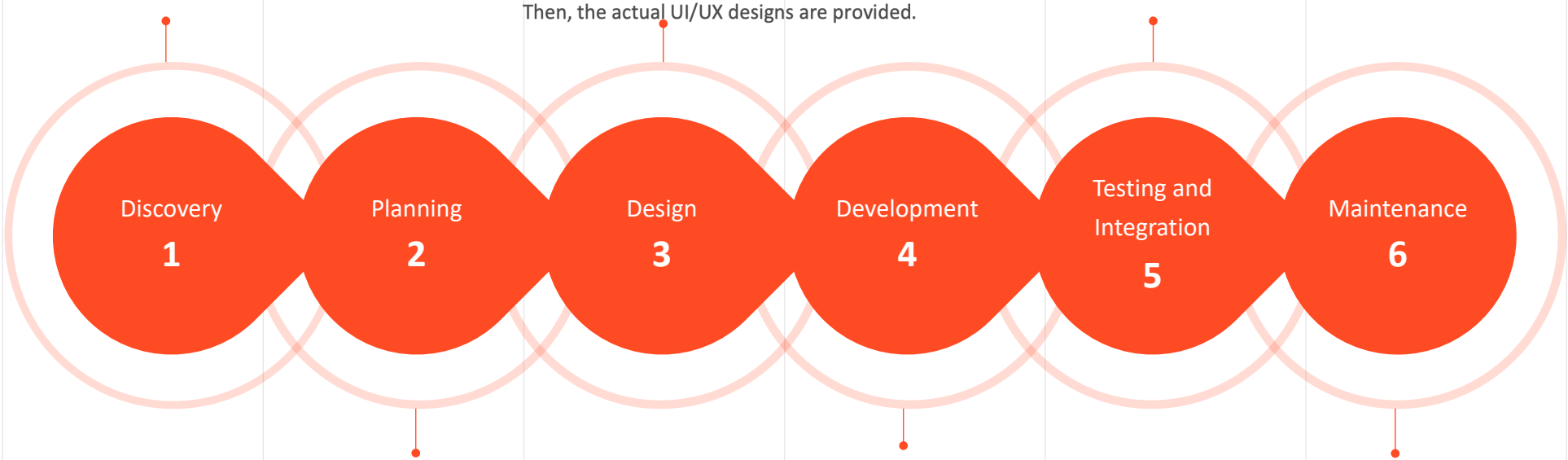
Collecting information and requirements and their analyses to ensure deep understanding of the goals, scope, potential risks and limitations.

Design

In this phase, requirements are being visualized, first through low-fidelity wireframes to get the basic idea of how the software will look like. Then, the actual UI/UX designs are provided.

Testing and Integration

Numerous tests are being conducted to ensure software is working according to specifications and there are no malfunctions.



Planning

Planning and documenting defined project deliverables, as well as processes and activities. Based on these, projected timelines and budget can be created.

Development

Development phase is the one where the system is constructed. Dev team, based on documentation and designs builds the components and assembles them into working increments.

Maintenance

After "going live", software is constantly monitored and tested to secure its performance and to act immediately in case of an error.

Phoenix Consultancy

Development Process

About Development Process

After detailed analyses had been conducted, to decide on the most appropriate architecture and technical stack, it has been agreed to proceed with:

- **Microservices Architecture** concept
- **.NET Core** framework for the services
- **Angular** for the User Interface

Phoenix Consultancy insists on code written in compliance with coding standards for the chosen framework and regular code reviews to ensure the code quality, while established deployment process provides successful integration of new components to existing platform. Dedicated QA team guarantees software quality and prevents errors by identifying mistakes or problems early during the development phase.

Coding Standards

Set of coding conventions for a specific coding language that recommend programming style, methods and practices that will be used while coding. Applying coding standards leads to Dev time optimization, minimal complexity, reduced risk of errors and easier debugging, simple maintenance, but also cost savings.

Code Review

The process of one or more people reviewing parts of the source code to prevent mistakes. By doing code review, code is created in a consistent manner, monitoring the quality, while Dev team collaborates and shares new techniques, expanding their knowledge and experience.

Deployment Process

The flow used for building, testing and integrating new code to an environment. Establishing deployment process reduces human error and saves time, leading to more frequent deployments by any team member.

Quality Assurance and Control

QA team defines and implements processes, standards and methodologies used to assure requested software quality is achieved.

Phoenix Consultancy

Coding Standards to be applied

Coding standards (recommendations) ensure the code is consistent, easily readable and understandable

01

Naming Standards

Naming standards provide consistent coding pattern and predictability. They represent the idea on how to name different entities of code. Some of the rules to be followed:

- Using one of the naming conventions: Pascal case, Camel case or Uppercase
- Abbreviations or contractions as parts of identifier names should not be used
- Use names that describe a type's meaning rather than names that describe the type
- Acronyms that are not generally accepted in the computing field should not be used
- Using verbs of verb phrases to name methods

02

Layout Standards

Code layout emphasizes the code structure and makes it easier to read. Few guidelines to be applied on the layout are:

- Writing one statement per line
- Writing one declaration per line
- If continuation lines are not automatically indented, they should be indented with one tab stop (equal to four spaces)
- Parentheses should be used to make clauses in an expression apparent

03

Commenting Standards

Comments should be written concisely, especially the ones referring to complex logic. The commenting recommendations are following:

- Comments should be placed on a separate line
- Comment text should begin with an uppercase letter
- Comment text should end with a period
- Between the comment delimiter (///) and comment text, one space should be inserted
- Formatted blocks of asterisks should not be created around comments

04

Language Standards

Language Standards describe practices to be followed while writing the code. Those practices include:

- Implicitly typed local variables
- Unsigned data types
- Event handling
- Static members
- LINQ queries

Phoenix Consultancy

Code Review

Code review can be done by one or few people to check for possible mistakes in code and ensure its quality. It can be done manually or using an automated code review tool.

The benefits of performing code review are maintaining the consistency in code writing between many team members, brainstorming and sharing techniques and code optimization, which ultimately leads to better software performance.

Most frequently used code review techniques are:

1. *Instant code review ("peer to peer")* - two developers simultaneously working on a code. One of them is writing the code while the other reads it and comments it along the way. It is considered quicker and efficient for developing complex programs, but also time and workforce inefficient.
2. *Ad hoc code review ("over the shoulder")* - after the developer has written the code, other developer is asked to review it. It is informal and more spontaneous way of reviewing the code. Its downside is high chance of glitches and missing mistakes in code.

Aside from these two, ***Meeting based and Tool based code reviews*** are commonly known, but they are less efficient and less used in practice.

Once the code review is successfully completed, the QA team takes the spotlight and starts their quality assurance process.

Deployment Procedures

About Deployment Procedures

Deployment procedure is a list of steps intended to guide the Dev team on how to successfully deliver any kind of software update to end users. These steps are followed every time new features, patches, modules, etc. are being released to a live environment.

Standard deployment procedure Phoenix Consultancy suggests is creating an automated deployment pipeline using Jenkins, as it provides great support for CI/CD (continuaous integration/continuous deployment).

Elements of the deployment pipeline are:

- Pull requests
- Builds
- Tests
- Blue-green deployment

Pull Requests

Pull request is a request for review of the changes before pushing them to a branch in the repository. It is usually created by the developer who had been working on the change. Once approved, pull request is being merged to the appropriate branch.

Builds

Build is a pre-release version of the software. Builds are created when a certain point in development has been reached or the code has been deemed ready for implementation, either for testing or outright release.

Tests

The goal of deployment testing is to make sure that build would work well once deployed. Testing the deployment scenario serves to measure system's stability, scalability and performance rates, but also to proactively identify issues that might occur while deploying.

Blue-green deployment

This deployment technique gradually transfers user traffic from previous software version to newly released version by running two identical production environments. In any point, only one of them is live. This type of deployment eliminates downtime during the deployment and reduces the risk by immediate rolling back to last version.

Phoenix Consultancy

Technology Stack

Web Application:

- Angular 9
- D3.js
- Angular Material
- Custom CSS/JS themes
- Mixpanel
- Authorize.NET

Infrastructure:

- AWS EC2 (Linux)
- Lambda Functions (.NET)
- API Gateway
- Application Loadbalancers
- AWS SQS

Storage:

- AWS RDS (Aurora)
- AWS DocumentDB (Mongo)
- AWS S3

Core Services:

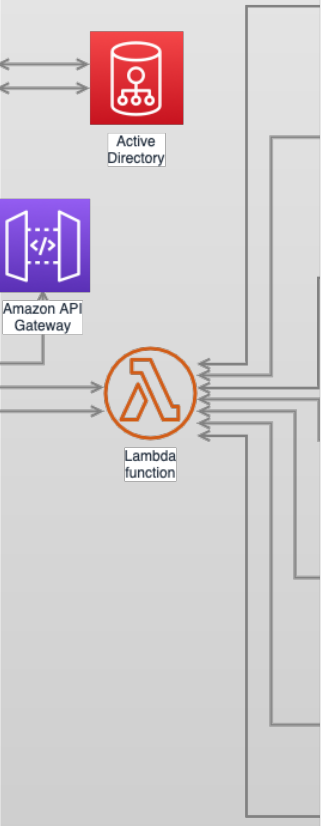
- .NET
- Entity Framework
- Diagnostics.HealthChecks
- NLog
- Swagger

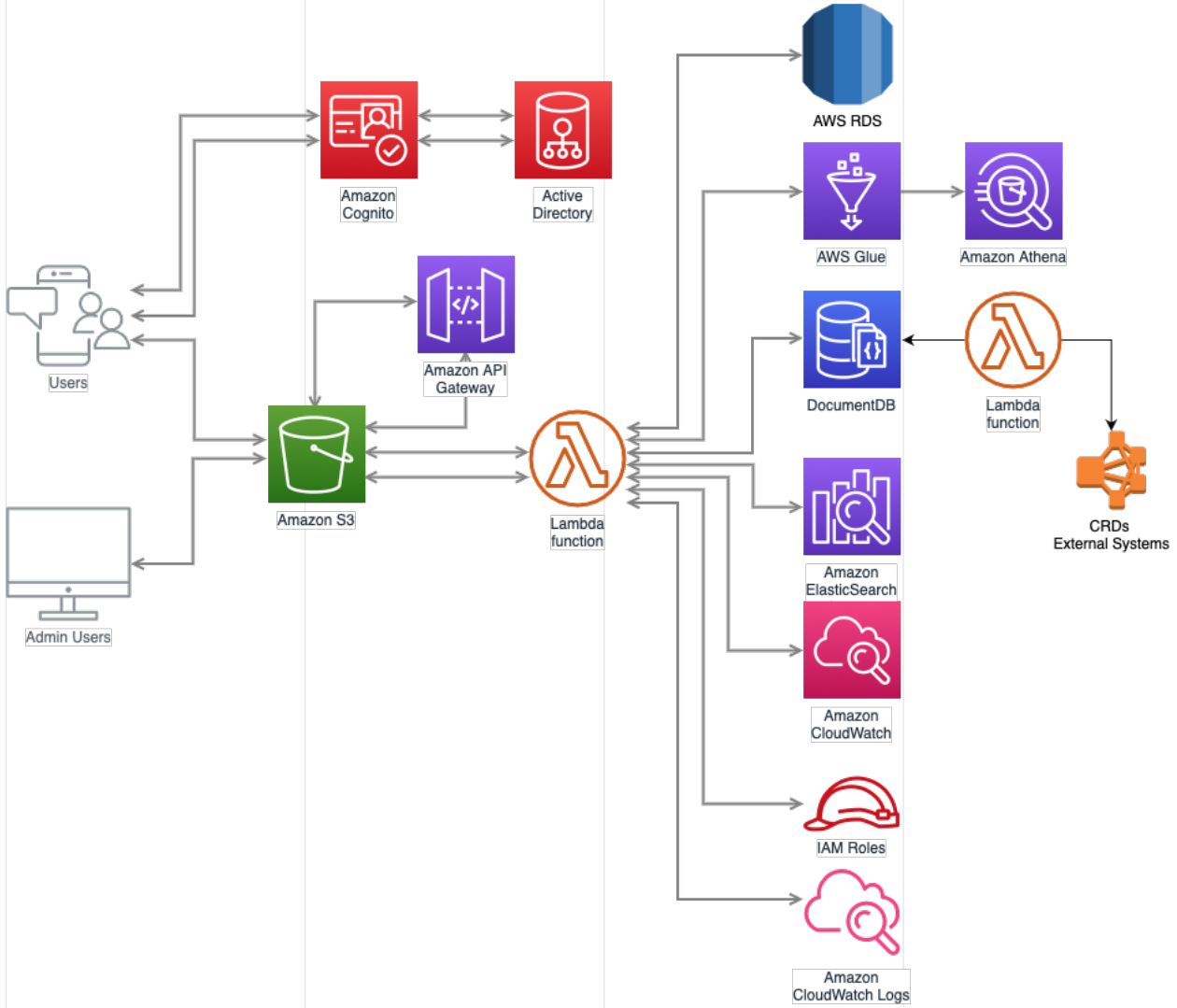
Phoenix Consultancy

Solution Architecture

Principles:

- Serverless
- Microservices
- Onion Architecture
- No single point of failure
- Redundancy
- Disaster Recovery environment
- Blue/Green Deployment





Phoenix Consultancy

Data Modeling

Data modeling process defines how data should be stored in the database, structures the data flow and emphasizes the important data relationships. Benefits of having data modeling properly set:

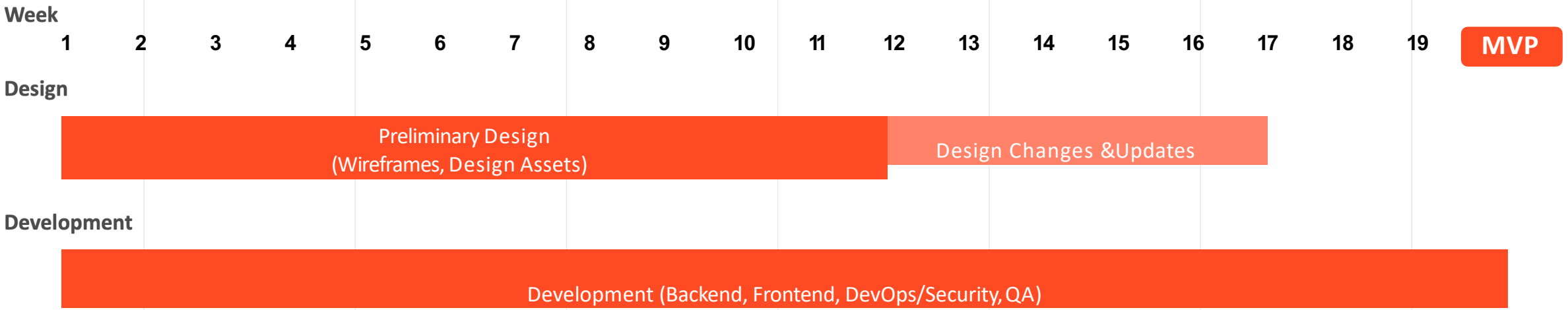
- Making sure all data objects are accurately represented before proceeding with development,
- Database can be easily designed at conceptual, physical and logical levels,
- Help in identifying missing or redundant data,
- Allows cheaper and faster infrastructure updates and maintenances.

The process of creating a data model includes these steps:

1. Identifying the entities
2. Identifying a key property
3. Drawing a rough data model draft
4. Identifying various data attributes
5. Mapping the attributes
6. Finalizing and validating the data model

Phoenix Consultancy

Timeline -20 Weeks



Project Timeline: 20 Weeks

Project Manager: 20 Weeks PT

Designer: 14 Weeks PT (30 hrs/week)

Lead Dev 1: 20 Weeks FT

Dev 2 (BE): 15 Weeks FT

Dev 3 (FE): 18 Weeks PT

Dev 5 (DevOps/Security): 11 Weeks PT

Dev 6 (QA Engineer): 15 Weeks PT

Notes:

- Design & Development run in parallel with 1-2 week sprints: Day 1 planning, Days 1-5 (10) execution, Day 5 (10) showcase
- Lead Dev splits time across stack
- Link: Backlog Estimates

Phoenix Consultancy