DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
THE UNIVERSITY OF TEXAS AT ARLINGTON

ARCHITECTURAL DESIGN SPECIFICATION
CSE 4316: SENIOR DESIGN I
SPRING 2023

ERS
EQUIPMENT RELIABILITY STRATEGIES

MATTHEW LUU
NICHOLAS NGUYEN
PABLO SANCHEZ
HOANG LONG QUAN NGUYEN
VIRAJ VIPINBHAI SABHAYA

REVISION HISTORY

Revision	Date	Author(s)	Description
0.1	03.15.2023	ML, NN, PS, HN, VS	document creation
0.2	5.10.2023	PS	complete draft

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

Equipment breakdown can cause delays in the production system. A vital machine breakdown may also cause the whole production line or the whole factory to be shut down completely, which can cost manufacturers hundreds of thousands of dollars per day. Moreover, it also can cause health, safety, and environmental accidents as well. According to research, around ninety percent of machine breakdown cases can be prevented with a good preventive maintenance plan. This is why Equipment Reliability Strategies Application was built for. It will help organizations, that own or operate process facilities, to manage all of their equipment in a professional and the most efficient way, to keep their equipment operating efficiently, minimize health, safety, and environmental accidents, and to avoid unexpected costly repairs in the future.

The Equipment Reliability Strategies is made available commercially, targeting organizations that own or operate process facilities across many industries. People in these organizations, such as managers, engineers, and inspection and maintenance staff are direct users of this application.

This application is a user-friendly web-based application, that uses the latest technologies for both front-end and back-end. To use this application users must create an account, and log in with authentication and access control. Based on their roles and granted permissions, the users will be able to create equipment, add or edit equipment properties, and calculate risks for each individual piece of equipment. From there, they will be able to create, update or delete, and manage default strategies, best practices, scenarios, actions, and mitigation, and create maintenance plans, inspection plans, and reports.

2   SYSTEM OVERVIEW

The Equipment Reliability Strategies Application consists of four layers, from top to bottom, which are the Web, Core, Model, and Database layers. These architectural "layers" are the top-level logical view, or an abstraction, of the design. Those four layers have related functionality associated with each other as shown in Figure 1 below.
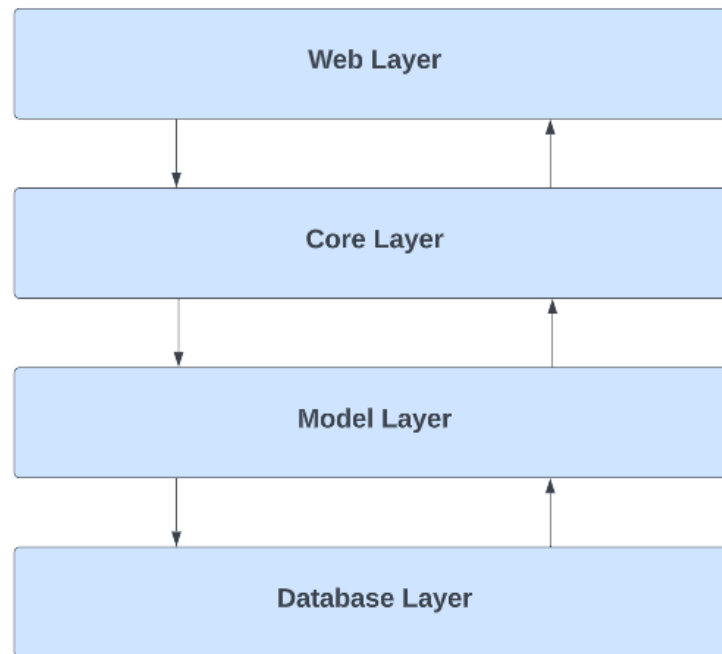

Figure 1: Architectural layer diagram

2.1   WEB LAYER

The Web layer is the outermost layer, which runs on the user's computer. It is a space where the users interact directly with this web application to complete business tasks. The Web layer will receive input data from the users, send requests to the server, receive responses from the server, and display the data from those responses to the users in a way that they can understand. The subsystems of this layer are Manage Default, Create Default, Add/Edit Properties, Show Equipment and Information, Create Equipment, Best Practices, Default Strategies, Login, Consequence Calculator, Create Account, and User Profile.

2.2   CORE LAYER

The Core layer contains the business logic of the application, including the security, access, and authentication tasks. This layer is responsible for performing specific business rules associated with the user requests from the Web layer. The subsystems of this layer are services and Authentication to interact with data.

2.3   MODEL LAYER

The Model layer is where the database turns into objects or a client-side representation of an object. Its main functions are connecting/accessing the database, retrieving requested data, and sending it back to the Core layer. The subsystems of this layer are business objects.

2.4 DATABASE LAYER

The Database layer is where all the database schema and application data are stored securely on the server side. Its functions are receiving requests from the Model layer, and sending responses including data back to it accordingly. The subsystems of this layer are SQL Server, and Project Scripts and Files.

3 SUBSYSTEM DEFINITIONS & DATA FLOW

This section shows a simple data flow diagram for this application, which includes four layers, their subsystems, and how the data flows between those layers and subsystems.
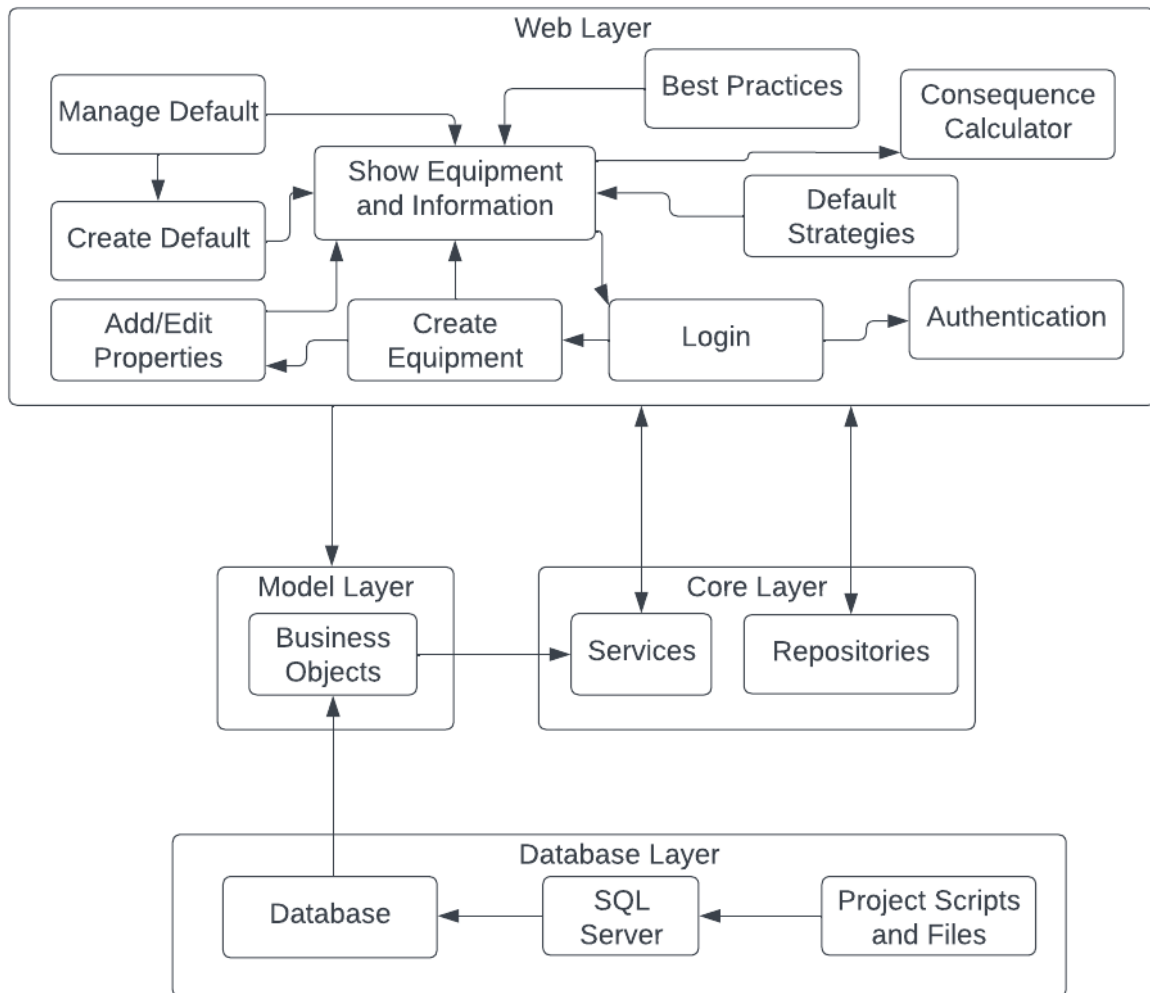

Figure 2: A simple data flow diagram

4 WEB LAYER SUBSYSTEMS

The Web Layer has the views, which is what is being presented to the User, helpers, language resources, security configurations, and the interact able parts of the web application. This layer will also mainly interact with the Core layer because it needs to know about the business objects which the Core layer can grab and give to the Web layer and has to be able to query the database which can be done with the Core layer as well.

4.1 LOGIN

The Login subsystem is used to log the user back into their specific account by using a previously created account and using the Core layer authentication subsystem to check if they used the correct login information.
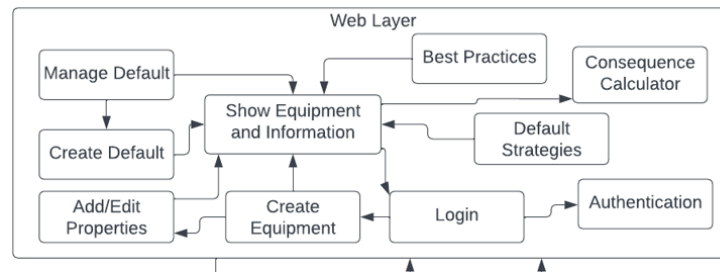
Figure 3: Web Layer Subsystem

4.1.1 ASSUMPTIONS

The assumption with this subsystem is that Becht will be using their own authentication to handle peoples login with the application.

4.1.2 RESPONSIBILITIES

It is responsible for prompting the user to enter the necessary information to log into their account and retrieve the correct user profile once they provide the needed information.

4.1.3 SUBSYSTEM INTERFACES

Table 2: Login Subsystem interfaces

ID	Description	Inputs	Outputs
#01	Profile retrieval	User credentials	User Profile

4.2 CREATE EQUIPMENT

The Create equipment subsystem allows users with proper access to create equipment that is being used within their units. It will use the Core layer for the authentication to make sure that the current user is allowed to create equipment for their unit.

4.2.1 ASSUMPTIONS

The assumption with this subsystem is that once a user has created new equipment it will then be saved to a database and shown to users within the same unit with the Show Equipment and Information subsystem.

4.2.2 RESPONSIBILITIES

It is responsible for making sure that the user has entered the necessary information for creating new equipment to be tracked by the application. It is also responsible for saving the created equipment to be able for it to be shown to other users.

4.2.3 SUBSYSTEM INTERFACES

Table 3: Create Equipment Subsystem interfaces

ID	Description	Inputs	Outputs
#01	Access Level Verification	User Profile	Accept/Refuse
#02	Equipment Creation	User Input	Equipment

4.3 ADD/EDIT PROPERTIES

The Add/Edit properties subsystem lets users update or add an equipment's properties so it can be displayed to everyone within the equipment's unit. It interact with the Service subsystem of the Core layer to be able to get the current properties of the equipment and either update them if they exist or add the new ones that were created.

4.3.1 ASSUMPTIONS

The assumption with this subsystem is that the created properties will be attached with its specific equipment and be saved with it in a database. It will also only allow certain users the ability to edit/add the properties of equipment.

4.3.2 RESPONSIBILITIES

It is responsible with making sure that the user has entered the necessary information for the specific equipment when creating the properties and editing them. It will also make sure that the user has the proper access to be able to change properties of equipment. It will lastly save the new properties with the corresponding equipment

4.3.3 SUBSYSTEM INTERFACES

Table 4: Add/Edit Properties Subsystem interfaces

ID	Description	Inputs	Outputs
#01	Access Level Verification	User Profile	Accept/Refuse
#02	Equipment Property	User Input	Update/New Equipment Property

4.4 CREATE DEFAULT

The Create default subsystem will allow for users to create defaults for certain types of equipment to be shown to the user. It will interact with the Core layer's services subsystem to view the equipment's properties to know its type and to then save the defaults into the database tied with a certain equipment type.

4.4.1 ASSUMPTIONS

The assumption with this subsystem is that once a user has created a new default it will then be saved to a database and shown to users within the same unit with the Show Equipment and Information subsystem with the specific equipment type.

4.4.2 RESPONSIBILITIES

It is responsible with making sure that the user has entered the necessary information for creating a new default to be tracked by the application. It is also responsible with properly maintaining the equipment type so that it can be shown with the correct equipment. It will also make sure that the user is able to create a default for an equipment type.

4.4.3 SUBSYSTEM INTERFACES

Table 5: Create Default Subsystem interfaces

ID	Description	Inputs	Outputs
#01	Access Level Verification	User Profile	Accept/Refuse
#02	Equipment Type Verification	Equipment Properties	Equipment Type
#03	Default Creation	User Input	Default

4.5 MANAGE DEFAULT

The Manage default subsystem allows the user to update an equipment's default whenever there is a change to it. It will also use the Core layer's Services subsystem to obtain the defaults so that they can be updated.

4.5.1 ASSUMPTIONS

The assumption with this subsystem is that the properties will in a database that the UI layer will be able to reach and that when the User is finished managing the defaults it will update the default within the database itself.

4.5.2 RESPONSIBILITIES

It is responsible with making sure that the user has entered the necessary information for the default and with checking that the user has the proper access level to be able to manage defaults.

4.5.3 SUBSYSTEM INTERFACES

Table 6: Manage Default Subsystem interfaces

ID	Description	Inputs	Outputs
#01	Access Level Verification	User Profile	Accept/Refuse
#02	Default Update	User Input	Updated Default

4.6 BEST PRACTICES

The best practices subsystem will obtain a list of best practices to be used on a specific type of equipment.

4.6.1 ASSUMPTIONS

The assumption with this subsystem is that the best practices will be provided beforehand to be used within the application and that they will be saved with the corresponding type of equipment within the database.

4.6.2 RESPONSIBILITIES

It is responsible with properly keeping the best practices with their corresponding equipment. It will also be responsible with obtaining the practices so that it can be used later.

4.6.3 SUBSYSTEM INTERFACES

Table 7: Best Practices Subsystem interfaces

ID	Description	Inputs	Outputs
#01	Equipment Type Verification	Equipment Properties	Equipment Type

4.7 DEFAULT STRATEGIES

The Default strategies Subsystem will obtain a list of default strategies to be used on a specific type of equipment.

4.7.1 ASSUMPTIONS

The assumption with this subsystem is that the default strategies will be provided beforehand to be used within the application and that they will be saved with the corresponding equipment within the database.

4.7.2 RESPONSIBILITIES

It is responsible with properly keeping the default strategies with their corresponding equipment. It will also be responsible with obtaining the default strategies so they could be used later.

4.7.3 SUBSYSTEM INTERFACES

Table 8: Default Strategies Subsystem interfaces

ID	Description	Inputs	Outputs
#01	Equipment Verification	Equipment Properties	Correct/Incorrect

4.8 SHOW EQUIPMENT AND INFORMATION

The Show equipment and information subsystem displays the equipment and its corresponding information to the user. It will interact with the Core layer's services subsystem to obtain all the necessary information that it will be trying to display

4.8.1 ASSUMPTIONS

The assumption with this subsystem is that it will show the same equipment in the beginning depending on the users unit and will be able to change its displayed sorting and filter out different things.

4.8.2 RESPONSIBILITIES

It is responsible with properly displaying the equipment and its information. It will also be responsible with checking what the user's unit is to display the correct equipment.

4.8.3 SUBSYSTEM INTERFACES

Table 9: Show Equipment and Information Subsystem interfaces

ID	Description	Inputs	Outputs
#01	Unit Verification	User Profile	User's Unit

4.9 CONSEQUENCE CALCULATOR

The Consequence Calculator will take in different factors that it will then use to try and calculate different types of risks like leak rate. It will use the Core layer's Services subsystem to obtain the factor types.

4.9.1 ASSUMPTIONS

The assumption with this subsystem is that there will be specific factors saved within the database.

4.9.2 RESPONSIBILITIES

The responsibilities of the subsystem are that it will produce a reasonable risk depending on different factors so the user will know what severity it could become.

4.9.3 SUBSYSTEM INTERFACES

Table 10: Consequence Calculator Subsystem interfaces

ID	Description	Inputs	Outputs
#01	Risk Calculation	Risk Factors	Calculated Risk

4.10 AUTHENTICATION

The Authentication subsystem handles security, access, and authentication tasks using strong, attested, scoped, public key-based credentials by web applications, for the purpose of strongly authenticating users.[1]

4.10.1 ASSUMPTIONS

A public key credential is created and stored by a WebAuthn Authenticator at the behest of a WebAuthn Relying Party, subject to user consent. Subsequently, the public key credential can only be accessed by origins belonging to that Relying Party. This scoping is enforced jointly by conforming User Agents and authenticators. Additionally, privacy across Relying Parties is maintained; Relying Parties are not able to detect any properties, or even the existence, of credentials scoped to other Relying Parties.[1]

4.10.2 RESPONSIBILITIES

The Authentication subsystem is responsible for handling security, access, and authentication tasks

4.10.3 SUBSYSTEM INTERFACES

Table 11: Authentication interfaces

ID	Description	Inputs	Outputs
#01	Authentication	public key credentials	user verified

5 CORE LAYER SUBSYSTEMS

The Core layer contains the business logic of the application, including the security, access, and authentication tasks. The Core layer uses the Model layer. It needs to know the business objects related to each service. The subsystems of this layer are Services and Authentication to interact with data.

5.1 SERVICES

The Services are part of an implementation of the Repository pattern that basically works as a link between the .NET project and the SQL Server database.
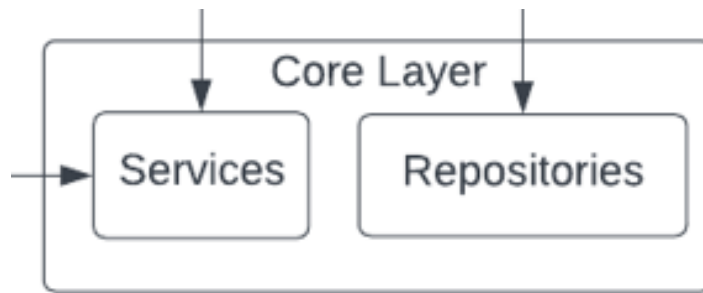

Figure 4: Core Layer Subsystem

5.1.1 ASSUMPTIONS

There is one service per business model in the project.

5.1.2 RESPONSIBILITIES

The Services are responsible for performing specific business rules associated with the user requests from the user interface layer.

5.1.3 SUBSYSTEM INTERFACES

Table 12: Services interfaces

ID	Description	Inputs	Outputs
#01	User requests	user inputs	business rules

5.2 REPOSITORIES

Repositories are also a part of an implementation of the Repository pattern that works as a link between the .NET project and the SQL Server database.

5.2.1 ASSUMPTIONS

There is one repository per database that is used.

5.2.2 RESPONSIBILITIES

It is responsible for retrieving the information that the services are trying to obtain and handing it off.

5.2.3 SUBSYSTEM INTERFACES

Table 13: Repositories interfaces

ID	Description	Inputs	Outputs
#01	Service Request	Service Requests	Data

6 MODEL LAYER SUBSYSTEMS

The model layer, sometimes referred to as the business layer is an essential part of software architecture that divides an application's business logic from its display layer and data layer. The business layer contains the application's core functionality, such as the rules and processes that govern the behavior of the system. It encapsulates the data and provides services to the presentation layer for interacting with the data. The business layer also handles business workflows, business rules, and data validation. By separating the business logic from the presentation and data layers, the business layer promotes a more modular, scalable, and maintainable software architecture. It also makes it easier to change the user interface or data storage without affecting the core business logic of the application.

6.1 BUSINESS OBJECTS

It includes the business objects, which are a 1:1 class representation of our database. A procedure known as scaffolding mechanically converts each database table into a C# class.
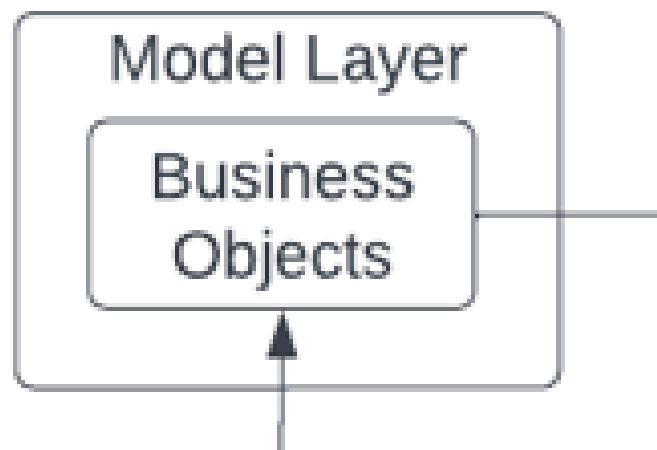

Figure 5: Model Layer Subsystem

6.1.1 ASSUMPTIONS

This subsystem is for creating our business objects. Each database table will be converted into C# classes that create our business objects.

6.1.2 RESPONSIBILITIES

Business objects in a business layer have the responsibility of representing in the software system the actual entities and operations of the business domain. Business objects serve as an abstraction layer between the presentation layer and the data layer by encapsulating the behavior and data of the application's business logic.

6.1.3 SUBSYSTEM INTERFACES

Table 14: Business Objects interfaces

ID	Description	Inputs	Outputs
#01	Scaffolding Database Table	Database Table	C# Class

7 DATABASE LAYER SUBSYSTEMS

In this section, the layer is described in some detail in terms of its specific subsystems. Describe each of the layers and its subsystems in a separate chapter/major subsection of this document. The content of each subsystem description should be similar. Include in this section any special considerations and/or trade-offs considered for the approach you have chosen.

7.1 SQL SERVER

Microsoft created the relational database management system known as SQL Server. Using a structured query language, it enables users to store, retrieve, and manage data (SQL). To meet the demands of modern businesses, SQL Server offers advanced functions including data warehousing, data analysis, and business intelligence.
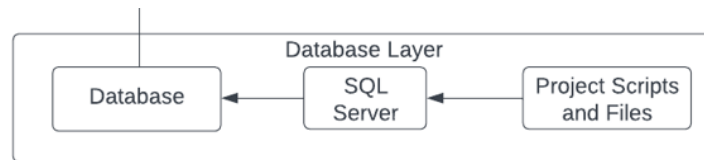

Figure 6: Database Layer Subsystem

7.1.1 ASSUMPTIONS

Users will be able to do CRUD operations. SQL server will directly interact with the database. SQL server will interact with the database tables to create business objects in the Model Layer. User Interface layer must be able to query the database.

7.1.2 RESPONSIBILITIES

This subsystem layer should enable users to store, retrieve, and manage data.

7.1.3 SUBSYSTEM INTERFACES

Table 15: SQL Server Subsystem interfaces

ID	Description	Inputs	Outputs
#01	Communicate with the back end to retrieve what the user has entered	User Input	None
#02	Communicate with the back end to output a variety of queries	Query	User Input

7.2 PROJECT SCRIPTS AND FILES

Project scripts and files are a collection of instructions or commands that are used to automate certain operations or processes inside a software development project.

7.2.1 ASSUMPTIONS

Users can assume developers can save time and reduce the risk of errors, as well as ensure consistency and repeatability in their processes with project scripts.

7.2.2 RESPONSIBILITIES

The subsystem layer contains .sql scripts and .xlsx files that create the SQL Scripts.

7.2.3 SUBSYSTEM INTERFACES

Table 16: Project Scripts and Files Subsystem interfaces

ID	Description	Inputs	Outputs
#01	SQL Scripts	.sql scripts .xlsx files	SQL Scripts

7.3 DATABASE

The database is what will hold what equipment there is, where it belongs to and the information to go with it.

7.3.1 ASSUMPTIONS

The assumption is that a database will already be provided and the application will just have to connect with it.

7.3.2 RESPONSIBILITIES

The database is responsible for saving any new equipment added by an authorized user and also making sure to save it with any other equipment that comes from the same area. It will also have to be able to give the application the equipment and its information whenever needed.

7.3.3 SUBSYSTEM INTERFACES

Table 17: Database interfaces

ID	Description	Inputs	Outputs
#01	SQL Server	Equipment and information	Table of information

REFERENCES

[1] Web Authentication: an API for Accessing Public Key Credentialslevel 2. W3C, W3C, 8 Apr. 2021, https://www.w3.org/TR/webauthn-2/.