

Resolve
Version 1.0

School of Dentistry

University of Washington, Seattle

Revision and Signoff Sheet

Change Record

Date	Author	Version	Change Reference
14 th March 2020	Palash Jhamnani	1.0	First Draft
1 st May 2020	Palash Jhamnani	1.0.1	

Reviewers

Date	Reviewer	Version	Comments
	Anya L. Levysmith	1.0	

Contents

Objective	4
Chosen Technology	4
Security	7
Authentication	7
Authorization	7
Data Layer (Models)	7
Model Descriptions	7
Entity Relationship Diagram	9
Business Logic Layer (Controllers)	10
Additional Case Types	10
Approval Workflow	10
Front End Layer (Views)	12

Objective

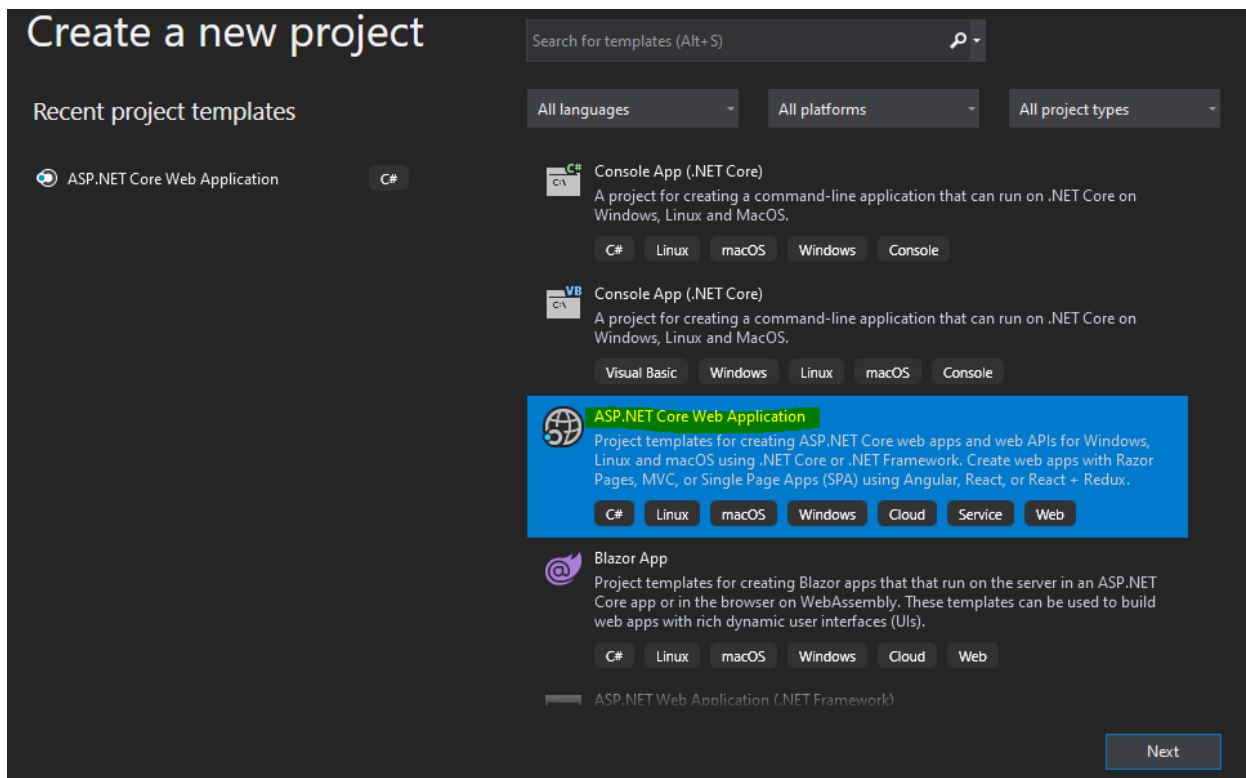
Resolve is a case management software developed to manage cases, which are service requests created by University of Washington (UW) community members in the UW School of Dentistry. These requests/cases then go through an approval workflow. Reporting dashboards would be created using the database entities for business reporting purposes.

GitHub Link: <https://github.com/palashjhamnani/Resolve.git>

Chosen Technology

Microsoft .Net Core Framework for building the web application

The chosen options have been highlighted in YELLOW in the screenshots below, while developing the application.



Configure your new project

ASP.NET Core Web Application

C#

Linux

macOS

Windows

Cloud

Service

Web

Project name


Resolve

Location

C:\Users\palas\source\repos

Solution

Create new solution

Solution name 

Resolve

☒ Place solution and project in the same directory

Back

Create

Create a new ASP.NET Core web application

.NET Core

ASP.NET Core 3.1



Empty

An empty project template for creating an ASP.NET Core application. This template does not have any content in it.



API

A project template for creating an ASP.NET Core application with an example Controller for a RESTful HTTP service. This template can also be used for ASP.NET Core MVC Views and Controllers.



Web Application

A project template for creating an ASP.NET Core application with example ASP.NET Razor Pages content.



Web Application (Model-View-Controller)

A project template for creating an ASP.NET Core application with example ASP.NET Core MVC Views and Controllers. This template can also be used for RESTful HTTP services.



Angular

A project template for creating an ASP.NET Core application with Angular



React.js

A project template for creating an ASP.NET Core application with React.js

[Get additional project templates](#)

Authentication

No Authentication

[Change](#)

Advanced

☒ Configure for HTTPS

☐ Enable Docker Support

(Requires [Docker Desktop](#))

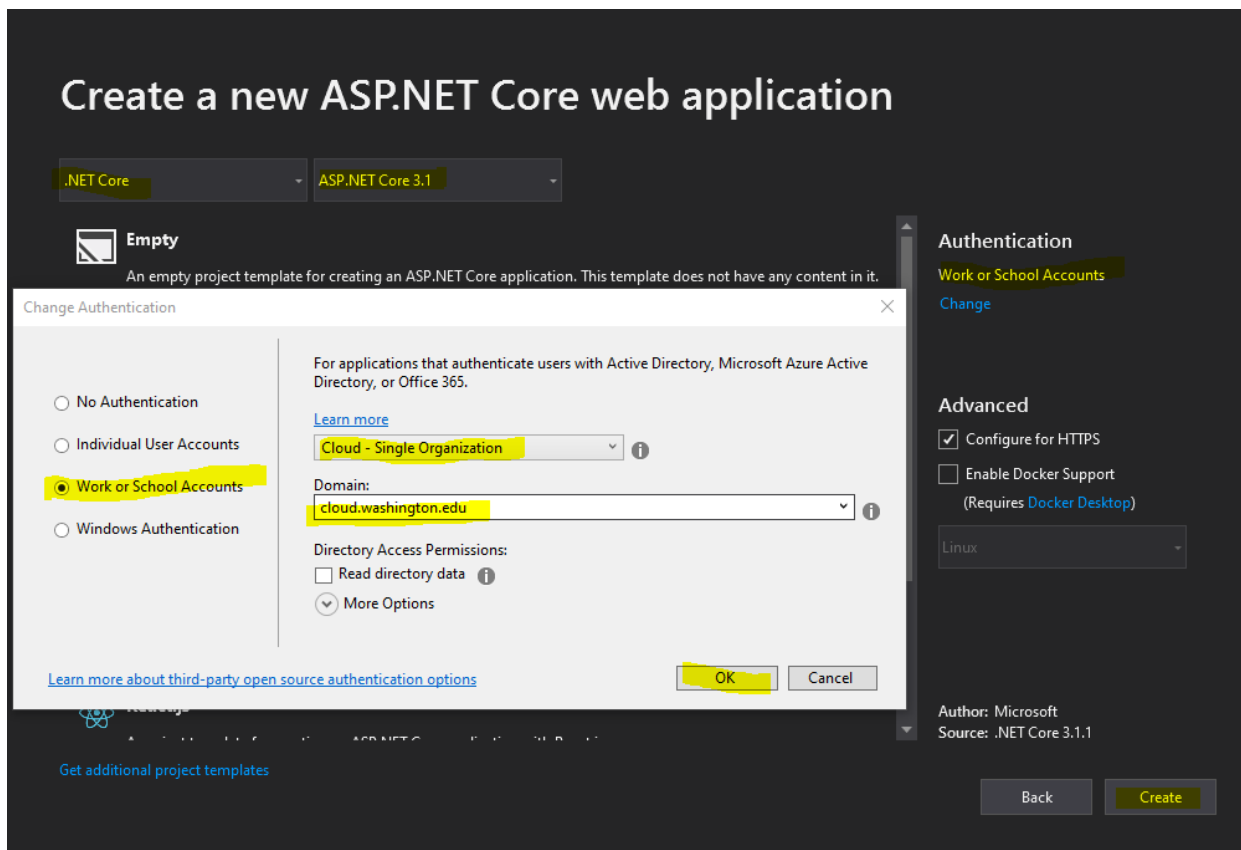
Linux

Author: Microsoft

Source: .NET Core 3.1.1

Back

Create



Link to authentication document for integrating UW Net ID

<https://itconnect.uw.edu/wares/msinf/authn/ldap/ldap-authentication-primer/>

<https://itconnect.uw.edu/wares/msinf/authn/ldap/>

<https://wiki.cac.washington.edu/display/infra/Shibboleth+for+UW+Web+Applications>

Security

Authentication

Authorization

Groups assigned to the application in Azure portal

Data Layer (Models)

Model Descriptions

CaseType: Sequential Approval? (Bit)

CaseTypeGroup:

Order: if Sequential Approval bit is on, order is checked before sending out emails for approval, else emails go out parallelly. ?

Entity/Model Description Table

No	Entity Name	Attributes	Why is Entity Included?	Relation with other Entities
Folder: Models				
1	LocalUser	<ul style="list-style-type: none">• FirstName• LastName• NetID• EmailID	Local User table is managed automatically. An entry in this table is created when a user logs in to Resolve for the first time. The data is picked from AzureAD	None
2	LocalGroup	<ul style="list-style-type: none">• LocalGroupID• GroupName• GroupDescription• FK: LocalUser	This table is managed manually by the admin. Groups are created as part of the application setup. ID comes from AzureAD (ObjectID)	One LocalUser is assigned to a LocalGroup as a default lead/approver
3	UserGroup	<ul style="list-style-type: none">• FK: LocalUser• FK: LocalGroup	This table is managed automatically. Entries are created as per User claims from AzureAD	Many-to-many mapping between LocalUser and LocalGroup
4	CaseType	<ul style="list-style-type: none">• CaseTypeID• CaseTypeTitle• LongDescription	This table is manually managed by the admin. Newer Case Types	GroupNumber is the the number of groups attached to this

		<ul style="list-style-type: none"> • GroupNumber 	can be added as required by the application users.	CaseType to whom the Case would be assigned.
5	Case	<ul style="list-style-type: none"> • CaseID • Auto: CaseCID • OnBehalfOf • CaseStatus • Auto: CaseCreationTimestamp • FK: CaseType • FK: LocalUser 	Case table is managed by the Users. An entry is created for every new case created by a user.	Many-to-One relationship with CaseType and LocalUser. A new Case can be of one CaseType and can be created by one LocalUser
6	OnBehalf	<ul style="list-style-type: none"> • PK, FK: Case • FK: LocalUser 	This table is managed by the Users. When a case is created by a user on behalf of another user, an entry is created in this table. One case can have only one entry; hence Case is PK.	One-to-One relationship with Case table, and Many-to-One relation with LocalUser table
7	CaseAttachment	<ul style="list-style-type: none"> • CaseAttachmentID • FK: Case • FK: LocalUser • FilePath • FileName • Auto: AttachmentTimestamp 	Managed by LocalUsers, an entry created for every new attachment, created for a Case by a LocalUser.	Many-to-One relation with Case table and LocalUser table. Many entries can be created for the same Case by many Users
8	CaseComment	<ul style="list-style-type: none"> • CaseCommentID • Comment • Auto: CommentTimestamp • FK: Case • FK: LocalUser 	Managed by LocalUsers, an entry created for every new comment, created for a Case by a LocalUser.	Many-to-One relation with Case table and LocalUser table. Many entries can be created for the same Case by many Users
9	CaseAudit	<ul style="list-style-type: none"> • CaseAuditID • Auto: AuditTimestamp • AuditLog • FK: Case • FK: LocalUser 	Managed automatically. An entry is created for every action taken by any user. A function call has to be made at code level by developer for data generation in this table	Many-to-One relation with Case table and LocalUser table. Many entries can be created for the same Case by many Users
10	GroupAssignment	<ul style="list-style-type: none"> • FK: Case • FK: LocalGroup 	By default, a Case is assigned to one group (as defined in CaseType table), however, if further group assignments are required, then an entry is created in this table.	Many-to-One relation with Case table and LocalGroup table. One Case can be assigned to multiple LocalGroups
11	CaseTypeGroup	<ul style="list-style-type: none"> • PK, FK: CaseTypeID • PK, FK: LocalGroupID • Approved • Order 	This table managed by Admins, for every CaseType's GroupNumber, entry to be created in this table.	Composite Primary key comprising of CaseTypeID and LocalGroupID
12	Approver	<ul style="list-style-type: none"> • FK: Case • FK: LocalUser • Approved • Order 	One entry is created in this table automatically for every new case created (Case -> CaseType -> DefaultGroup -> DefaultApprover). However, more approvers can be added	Many-to-One relation with Case table and LocalUser table. One Case can be assigned to multiple LocalUsers

			by the admin or the pre-assigned approver.	
Sub Folder: CaseTypeModels				
13	SampleCaseType1	<ul style="list-style-type: none"> • PK, FK: Case • Attribute1 • Attribute2 • ... 	To add specific details for a Case Type, new entities can be created by the admin/developer as required by the users. For every new Case Type added, a new model must be created under the “~/Models/CaseTypeModels/” folder, if specific details are required for that CaseType	One-to-One relation with Case table. Only one entry can be created in this table per Case.
14	SampleCaseType2	<ul style="list-style-type: none"> • PK, FK: Case • Attribute1 • Attribute2 • ... 		
	...	<ul style="list-style-type: none"> • PK, FK: Case • ... 		

PK: Primary Key

FK: Foreign Key

Auto: Automatically Generated by Database on addition of a row

Database used: Microsoft SQL Server

Below is the connection string specified in the *appsettings.json* file.

```
"ConnectionStrings": {
  "MvcMovieContext":
"Server=.\SQLExpress;Database=Resolve;Trusted_Connection=True;MultipleActiveResultSets=true"
}
```

Entity Relationship Diagram

Business Logic Layer (Controllers)

Additional Case Types

Steps to add a new case type:

- Create a model under "~/Models/CaseTypeModels/{CaseTypeTitle}.cs"
- P.S - Name of the model must be same as "CaseTypeTitle" from CaseType entity
- Add entry to "~/Data/ResolveCaseContext.cs"
- Add entry to "~/Models/Case.cs"
- Add controller under "~/Areas/CaseSpecificDetails/Controllers/{CaseTypeTitle}Controller.cs"
- Add view under "~/Areas/CaseSpecificDetails/Views/{CaseTypeTitle}/Index.cshtml"
- Add view under "~/Areas/CaseSpecificDetails/Views/{CaseTypeTitle}/Create.cshtml"
- Add an entry to "Details" action in "~/Controllers/CasesController.cs" to include details of new Case Type.

Approval Workflow

In the **Approver** model, the 'Approved' attribute can have 3 values as described below:

'Approved' Attribute Value	Significance
-1	Disapproved/Rejected
0 (Default)	Neutral (Newly added approver)
1	Approved/Accepted

When an approver approves a case or rejects a case, the value is set automatically in the 'Approved' attribute.

Below are the business rules programmed for Approval Workflow:

For a case to be deemed approved, it needs to have an approval by ALL assigned approvers, or by the admin, on behalf of all assigned approvers. Even if there is a single reject, the Case is deemed to remain in rejected status.

- When a Case is created, a default approver is assigned to it, i.e. an entry in the **Approver** model is created.
- This default approver comes from the flow: Case -> CaseType -> Default LocalGroup -> DefaultApprover.
- Every Case will belong to a CaseType, every CaseType will have a Default LocalGroup associated with it, and every LocalGroup will have a default Approver (LocalUser) associated with it.
- Multiple approvers can be assigned to a Case.
- An Admin, or an existing Approver can add additional approvers for the Case. For other users, the option to add additional approvers will remain disabled.
- Admins can approve any Case. When an Admin approves a Case, the value for 'Approved' attribute for all approvers will be set to 1 and the Case will be marked as Resolved.
- Once the case is approved by all assigned approvers or the admin, the buttons to Approve/Reject get replaced by Reopen, and a badge with "Resolved" keyword is put for the case.
- Until the Case is resolved, a badge with "# Approvals Pending" is put for the case.

Logic for managing approvals:

```
if user is approver or admin:
    if current_user is approver AND admin:
        if case processed by all:
            show reopen for all
        elif case processed by self but pending on others:
            show approve/reject buttons to process on behalf of all

    elif current_user is only admin:
        if case approved by all approvers:
            show reopen button for all approvers
        else:
            show approve/reject buttons to approve/reject on behalf of all
    approvers

    elif current_user is only approver:
        if case approved by self:
            show reopen for self
        else:
            show approve/reject buttons for self

else:
    show nothing but audit
```

Front End Layer (Views)