

# IMO Maritime Single Window



- *a generic multi-platform Open Source Maritime Single Window developed in the Norwegian and Antigua and Barbuda MSW project.*

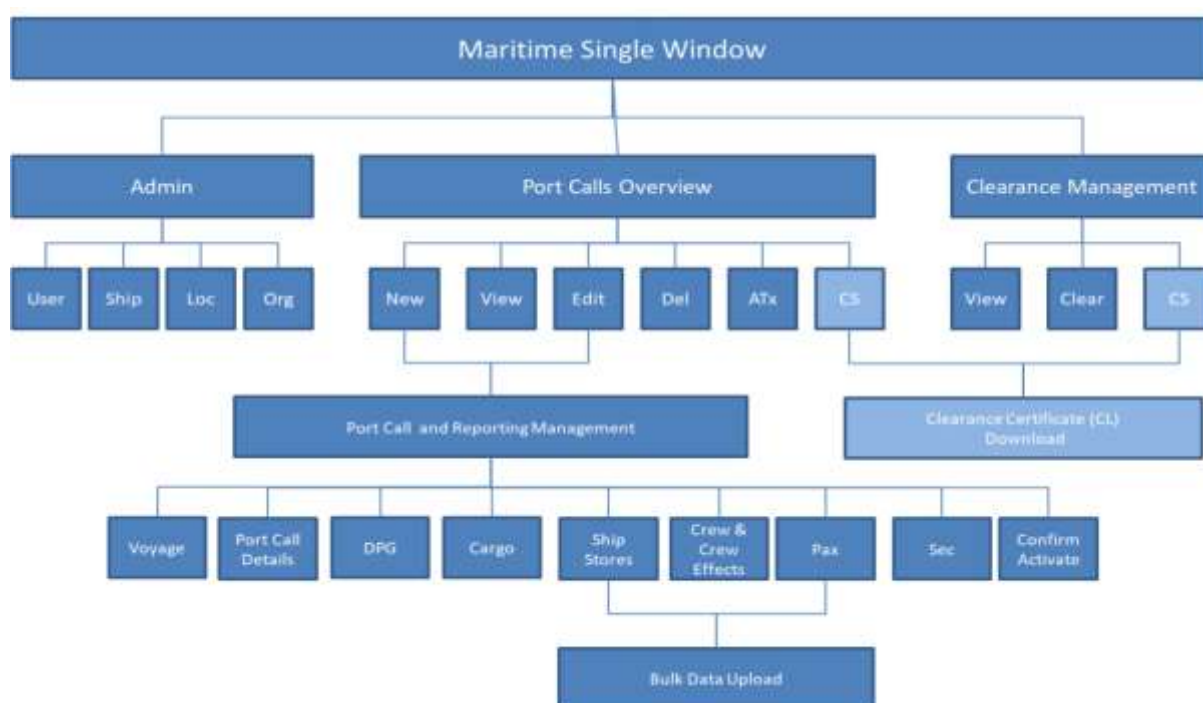
## Introduction

1. This document gives the reader a look under the hood of the Maritime Single Window vehicle developed in the IMO Maritime Single Window project (2017-2019).
  2. The generic Single Window system and its environment, including the Client, Server, Database and Middleware are as much as possible developed in the spirit of the Open Source Initiative. Also the system is developed to support cross-platform implementations thus the generic Maritime Single Window system could be deployed on multiple computing platforms.
  3. The generic Maritime Single Window developed in the Antigua and Barbuda and Norway project is a single-page web application (SPA/Client) which interacts with the user by dynamically, rewriting the current page rather than loading entire new pages from a server. This approach avoids interruption of the user experience between successive pages, making the application behave more like a desktop application. In the application the appropriate resources are dynamically loaded and added to the page as necessary, usually in response to user actions. The page does not reload at any point in the process, nor does control transfer to another page, although the location hash or the HTML5 History API can be used to provide the perception and navigability of separate logical pages in the application.
  4. The application is mainly a web application that requires manual input, but has a built in capacity to upload bulky data, such as Crew Lists. In principal all information is electronic and digitized, but the application allows for a download of the Clearance Certificate (CS) to prevent any misunderstandings and interruptions in the next port of call. Interaction with the single page application involves dynamic communication with the web server and database behind the scenes.
-

## The Client application architecture

5. In principal the generic Maritime Single Window has three main functions or modules;
- I. **Administrative functions**, such as maintaining the UserID' of the system, Ship Data maintenance, and interfaces to maintain Organizations and Locations
  - II. The main module of the application is the **Port Calls Overview**. The interface that lets the reporting party manage the existing port call, create a new call or simple view the details registered.
  - III. The third interface, **Clearance Management** allows the authorities to view and give Clearance to a particular ship having a port call. The Authorities and the reporting party share the function to view or download the Clearance Certificate.

A complete overview of the application architecture is shown in the following figure.



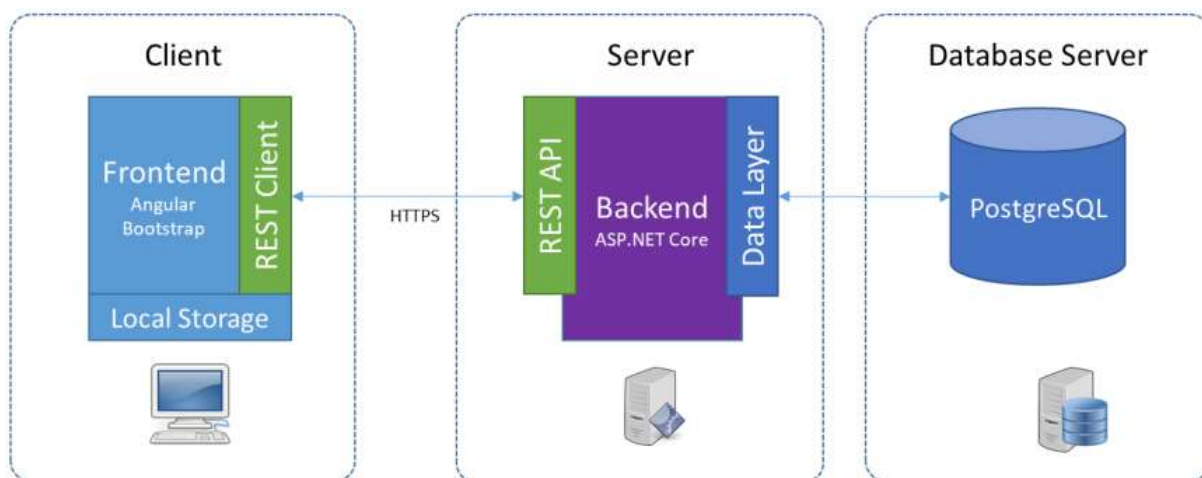
6. As seen on the figure, most of the Single Window reporting functionalities sit within the **Port Call and Reporting Management** module. This is where the main mandatory reporting obligations are registered. The generic application covers all the required information in the current FAL forms and the Security reporting. The registrations forms are optimized to keep the reporting work to a minimum i.e the Crew Effects List is only one simple field of input, and is found on the Crew List tab in the application.
7. To facilitate the reporting of the detailed Ship Store, Crew- and Passenger list, upload functionality has been developed to allow for a possible upload of data in a pre-populated spreadsheet template<sup>1</sup>.

<sup>1</sup> See Annex for more information on the spreadsheet template and functionalities

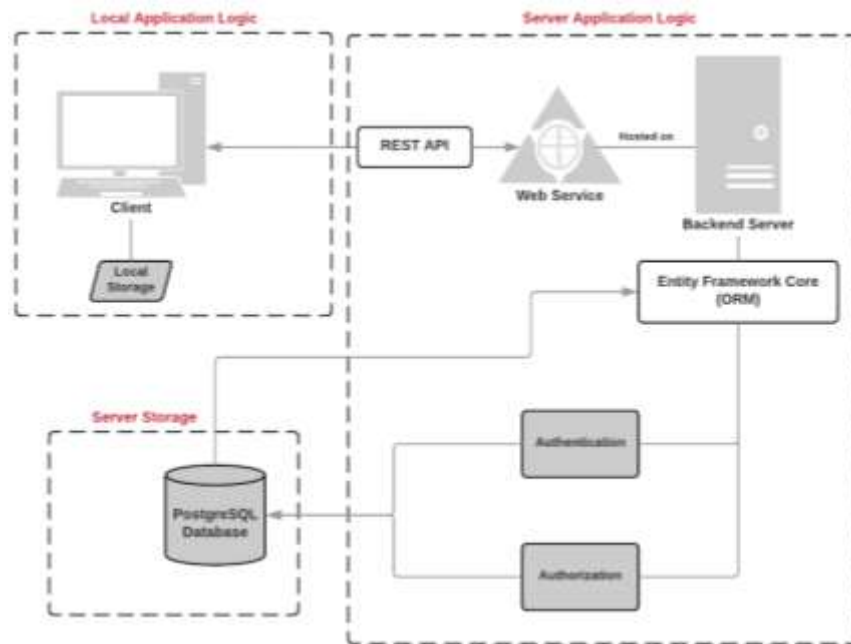
8. A newly created Port Call will be stored in *Draft Mode* and only available for the Reporting Party until it is *Confirmed and Activated*. At this point the Port Call become official and made available for the Authorities for further actions and Clearance.
9. The Clearance Certificate (CS) is a function that is shared between the Reporting Party and the Authorities. This function is only activated after all the Clearance has been finalized. After Clearance is done electronically a copy of the Clearance Certificate populated with key Port Call Information can be downloaded for further required manual processes.

### The generic Maritime Single Window application architecture

10. The Maritime Single Window has a distributed application structure through standard client-server and database architecture.



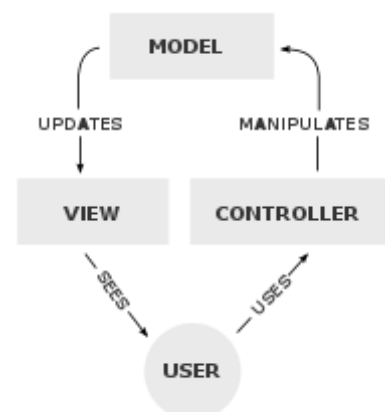
11. The Client (frontend) is developed using *Angular* which is a structural framework for developing dynamic web apps. To have a responsive behavior suitable for mobile devices, HTML and CSS-based design templates and interface components the web framework *Bootstrap* is used as a part of the Client platform.
12. To enable a cross-platform, high-performance open-source framework, the Server (backend) is developed using an *ASP.NET Core* framework. The communication between the Client and Server are encrypted using *Hyper Text Transfer Protocol Secure* (HTTPS).
13. The database server is based upon a *PostgreSQL* database which is a powerful, open source object-relational database system that uses and extends the SQL language combined with many features that safely store and scale the most complicated data workloads.
14. The REST-endpoints in the client and server typically refers to some object or set of objects (i.e; *api/portcall/user*) are developed using *C#* (C Sharp) and utilizes the *Entity Framework Core* (EF Core). The Object-Relational Mapping (ORM) in the EF Core translates objects to and from the relational structure of the *PostgreSQL* database. The EF Core is lightweight, extensible and support cross platform development. In the application the EF Core is also used for access control.
15. The figure below shows an even more detailed visualization of the system components and the communication between the various parts and components.



## The Software project

16. The software project is structured with a Model–view–controller (MVC) architecture. This architecture divides the application into three interconnected parts. This is done to separate internal representations of information from the ways information is presented to and accepted from the user.

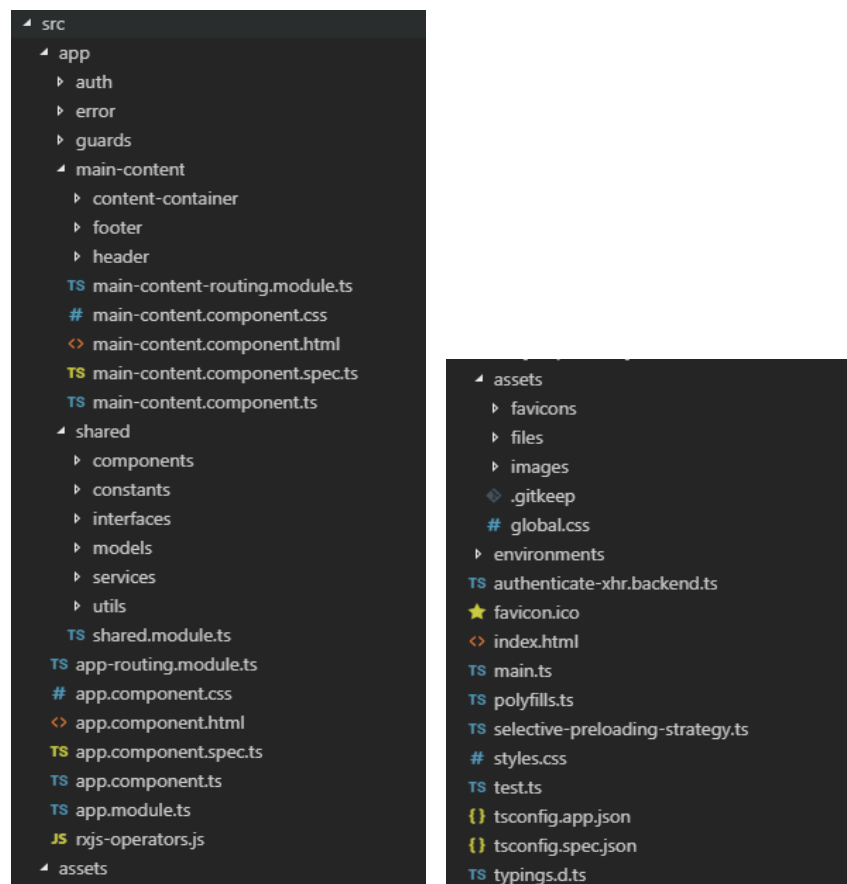
17. The MVC design pattern decouples these major components allowing for efficient code reuse and parallel development. The model is the central component of the pattern. It is the application's dynamic data structure, independent of the user interface. It directly manages the data, logic and rules of the application.



18. A view can be any output representation the various parts of the Single Window application. The controller accepts input and converts it to commands for the model or view. In addition to dividing the application into three kinds of components, the model–view–controller design defines the interactions between them.

## Client project (frontend)

19. The client follows the recommended structure of the Angular framework and is generated using the Angular CLI. The figure below shows the structure of the Client.



20. There are five main directories in the client project;


- **auth**  
This directory contains the client side of all the login, authentication, guards, password management amongst other.
- **error**  
This directory contains the module for error handling in the client.guards
- **guards**  
This directory contains the root and error guards of the client. These guards protect the routes within the client.
- **main-content**  
This directory contains most of the components used in the client. The directory is divided into three subdirectories; footer, header and content-container.
  - Footer and header contain the components for these two features of the client.
  - Content-container contains all the other components in the client divided into subdirectories based upon where in the client they are used.
- The three main directories are; account, basis-data and port-call.

- The account directory contains components to do with the currently logged in account.
- The basis-data directory contains components related to organization, ships, users and locations.
- The port-call directory contains components for clearance, confirmation, overview, registration and view-port-call.
- **Shared**  
This directory contains all the components that are often shared between modules/components. Examples include such components as date-picker, country-select, search bars, buttons. Constants, interfaces, models, services and utilities are also located within this directory.

21. The client project is developed using Angular 5 and TypeScript (superset of JavaScript). The Client project is compiled down to a regular JavaScript file which is moved into the “*wwwroot*” directory on the server project and will comprise the code that is executed on the web site.

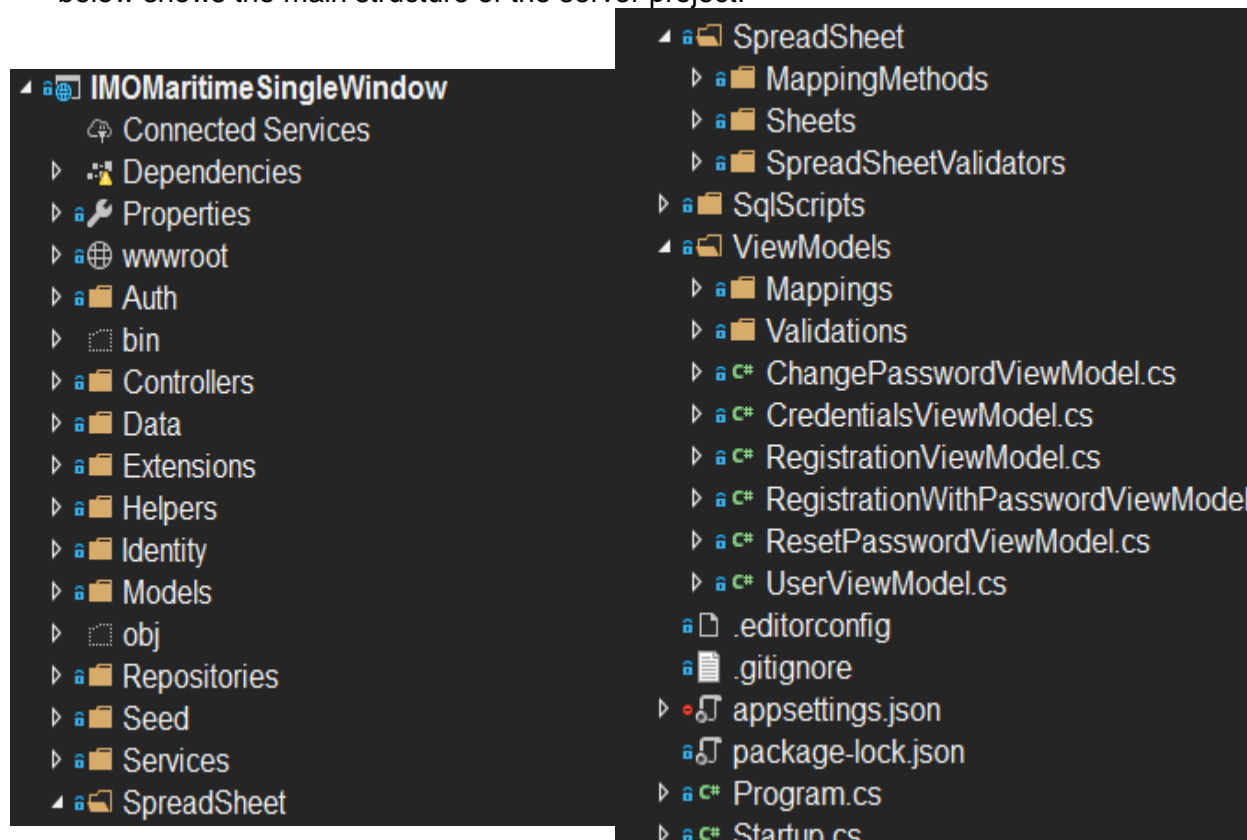
22. The Client project is only used for development and will have no impact on the product until it is compiled and moved into the *wwwroot* directory.

The prerequisites and steps to compile the client are described in the *README* file in the GitHub repository.

Branch: master ▾		New pull request	Create new file	Upload files	Find file	Clone or download ▾
bardlokas Build Enabled Prod						Latest commit eaa21c1 8 days ago
📁 .vscode	Roles listed in user_reg. Auth headers set before request. Injectables					10 months ago
📁 IMOMaritimeSingleWindow.Tests	Release 13-12-2018 (#54)					2 months ago
📁 IMOMaritimeSingleWindow	Build Enabled Prod					8 days ago
📄 .gitattributes	Add .gitignore and .gitattributes.					a year ago
📄 .gitignore	Simple start of cargo fal form, with smart table					6 months ago
📄 BuildFrontend.bat	Build front end					6 months ago
📄 CONTRIBUTING.md	Create CONTRIBUTING.md					15 days ago
📄 IMOMaritimeSingleWindow.sln	Made testing available from Visual Studio					11 months ago
📄 LICENSE	Update LICENSE					a month ago
📄 README.md	Update README.md					15 days ago
📄 package-lock.json	Basic start for component to edit location data.					8 months ago
📄 README.md						

## Server project (backend)

23. The server code follows the recommended structure for MVC projects, and the figure below shows the main structure of the server project.



24. The following table describes the top folders for the server project.

Folder	Description
Auth	Classes that handle authentication/authorization and Web Tokens
Controllers	Contains controllers which are the connection between the client and the server. The RESI-endpoints are defined here.
Data	Classes that handle access to data and ORM-mappings
Extensions	Extension methods that expand class-functionality
Helpers	Constants, mappings, services
Identity	Contains classes to interact with the ASP .NET Core Identity framework
Models	Contains models that represent entities in the database
Repositories	Classes with logic to communicate with the relevant storage medium
Services	Contains services (e.g EmailSender)
SpreadSheet	Contains the classes for SpreadSheet import. Contains mapping, validation, definitions.
SqlScripts	Contains the database scripts to be run on releases
ViewModels	Contains classes that describe the objects sent between the client and the server.
Wwwroot	Contains the compiled client project and is ready to run



25. The server project hosts a REST-API with various methods located in controller classes, these classes include but are not limited to

Controller	Description
Account	Contains methods for creating/modifying accounts as well as various administrative methods such as deactivating users.
Auth	Contains methods for authenticating users (Login, credentials etc)
Claim	Retrieves claims to be used in JWT.
Connection	Retrieves current database connection state.
Country	Contains various methods for retrieving countries (by exact name, by id or search)
Dpg	Retrieves DPG items from the database.
File	Contains method for uploading and importing data from an excel sheet. Currently implemented methods for Crew, Passengers and Ship Stores.
IAccount	Interface Implementation of the Account Controller
Location	Contains methods for registering, updating and retrieving location data.
PersonOnBoard	Retrieve Crew/Pax from the database
Purpose	Contains methods related to purpose of a port call.
Test	Contains various test methods

26. ShipController offers amongst other these endpoints

Endpoint (protocol)	Description
api/ship (post)	Register a ship
api/ship (put)	Update ship
api/ship/{shipId}/internationalShipSecurityCertificate/isscId (put)	Update the ISSC of the ship with the shipId
api/ship/search{searchTerm}/{amount} (get)	Returns a list of ships based on search criteria supplied from request.
api/ship/search{searchTerm}/{amount}/{enumValue} (get)	Returns a list of ships based on search criteria filtered on shipType(enumValue)

27. PortCallController offers amongst other these endpoints

Endpoint (protocol)	Description
api/portcall/{portCallId}/falShipStores (get/put)	Retrieves/Updates ship stores for a specified port call
api/portcall{portCallId}/consignments (get/put)	Retrieves/Updates consignments for a specified port call
api/portcall{portCallId}/personOnBoard (get/put)	Retrieves/Updates Person on board for a specified port call
api/portcall{portCallId}/personOnBoard/personOnBoardType/{EnumValue} (get)	Retrieves all Persons on Board by portcallId and type (crew/pax)
api/portcall{portCallId}/personOnBoard/personOnBoardType/{EnumValue} (put)	Updates person on board list for a specified port call (portCallId) for a specific type (crew/pax)
api/portcall/user (get)	Gets all port calls related to the currently logged in user
api/portcall/updatestatus/awaitingclearance/{portCallId} (post)	Sets the status for a specified port call to "Awaiting Clearance"
api/portcall/updatestatus/cleared/{portCallId} (post)	Sets the status for a specified port call to "cleared"



api/portcall/updatestatus/completed/{portCallId} (post)	Sets the status for a specified port call to "completed"
api/portcall/updatestatus/cancelled/{portCallId} (post)	Sets the status for a specified port call to "cancelled"
api/portcall/updatestatus/draft/{portCallId} (post)	Sets the status for a specified port call to "draft"
api/portcall/delete/{portCallId} (post)	Sets the status for a specified port call to "deleted"
api/portCall (post)	Register a new port call
api/portCall/{id} (get)	Gets a specific port call by ID

## Database

28. The project uses an object-relational database model using the open source relational database management system (RDMS) PostgreSQL.

The following tables are contained within this database

1	organization	21	role	40	identity_document	61	department
2	international_ship_security_certificate	22	user_token	41	security_level	62	customs_cargo
3	port_call_details	23	password	42	ship_flag_code	63	customs_cargo_type
4	organization_type	24	role_claim	43	ship_hull_type	64	identity_document_type
5	fal_security	25	claim_type	44	ship_type_group	65	__EFMigrationsHistory
6	port_call	26	person	45	ship_source		
7	location	27	claim	46	ship_status		
8	dpg_on_board	28	port_call_purpose	47	ship_power_type		
9	measurement_type	29	user	48	ship_length_type		
10	security_previous_port_of_call	30	ship_contact	49	ship_mmsi_mid_code		
11	ship_to_ship_activity	31	port_call_has_port_call_purpose	50	ship_history		
12	company_security_officer	32	ship_type	51	ship		
13	person_on_board_type	33	ship_certificate	52	port_call_status		
14	cargo_item	34	imo_hazard_class	53	marpol_category		
15	gender	35	ship_certificate_type	54	organization_port_call		
16	package_type	36	county	55	country		
17	fal_ship_stores	37	ship_breadth_type	56	location_type		
18	certificate_of_registry	38	person_on_board	57	location_source		
19	consignment	39	dpg_type	58	municipality		
20	user_login	40	identity_document	59	contact_medium		
				60	dpg		

29. The MSW project has deployed a public site of the Single Window System. The database for this site is hosted on a virtual Linux machine in the Azure cloud portal. To connect to the database from the Server project the file *appsettings.json* is used. The listing below shows the connect parameters for this particular deployment. For obvious reasons most of the information is left out in the figure below.

```
{
  "ConnectionStrings": {
    "OpenSSN": "User
ID=postgres;Password=XXXX;Host=xx.xxx.xx.xx;Port=xxxx;Database=xxx
xx;keepalive=60;",
    "UserDatabase": ""
  },
  "AppSettings": {
    "Secret": "xxxxxxxxxxxx"
  },
  "JwtIssuerOptions": {
    "Issuer": "xxxx",
    "Audience": "https://imo-msw-public-test.azurewebsites.net/"
  },
  "SendGridOptions": {
    "ApiKey": "xxxxx"
  },
  "EmailSenderOptions": {
    "From": "noreply@imo-msw.org"
  }
}
```

## Security

30. The information in the Maritime Single Window is considered too be sensitive and security is therefore a very important element. In a local installation security will normally mean to setup firewalls and various other means of protecting the computer site and data. However the application also has to have built in mechanisms to prevent security breached and malicious attacks.
31. The Client communicates with the Server using the internet, thus the server requires the Client to communicate using a cryptographic protocol. By using *The Transport Layer Security* (TLS), the Maritime Single Window network establish secure communication.
32. Access to the system requires that a systems administrator registers the user account and assign a role for the particular user. The newly created user will receive an email to create a new password for the account. The password registered by the new user is hashed using a pseudo-random salt. The system uses *Password-Based Key Derivation Function 2* (PBKDF2) to reduce the vulnerability of encrypted keys to brute force attacks.
33. PBKDF2 is used with the following parameters as standard:
  - Salt: 128 bits
  - Derivated key: 256 bits
  - Iterations: 1000
  - Algorithm: HMAC-SHA1

34. After a successful authenticated login, the userID is provided a *JSON Web Token* (JWT , RFC 7519) which contains information that the server uses to authenticate the userID sending the request. Below is a typical a JWT token with a Keyed-hash Message Authentication Code HS256:

```
{
  "auth_token":
  {
    "header":
    {
      "alg": "HS256",
      "typ": "JWT",
    },
    "payload":
    {
      "claims": [],
      "aud": ,
      "exp": ,
      "iat": ,
      "id": ,
      "iss": ,
      "jti": <JWT id>,
      "nbf": ,
      "rol": ,
      "sub": ,
    },
    "signature": ,
  }
}
```

35. Each attribute in for the object “auth\_token.payload” is called a “claim”. The claim “auth\_token.payload.claims” contains application specific rights for the relevant userID. The other claims are called “registered claim names”. The attribute “header” specifies what type it is, JWT, and what algorithm is used to generate the signature, HS256.
36. The Authorization library from AspnetCore is used to protect the REST-Endpoints by annotating the various endpoints with authorization levels.
37. The JWT is stored in the LocalStorage of the browser which in turn hinders Cross Site Request Forgery attacks which prevents an attack that forces an end user to execute unwanted actions on the Maritime Single Window session in which they are currently authenticated.
38. The *Language-Integrated Query* (LINQ) framework is used to protect from SQL injection. LINQ performs parameter-based SQL queries which prevent a malicious attacker from injecting nefarious SQL statements through inputs in the client.
39. Strings to connect to the database (username, password and URL) are retrieved from a config file located on the server. This file is not in the source code and not accessible from the outside. The same principle applies for the key to sign tokens.

## **Source Code and public site**

40. The source code can be found on the following GitHub Repository:

[https://github.com/Fundator/IMO-Maritime-Single-Window'](https://github.com/Fundator/IMO-Maritime-Single-Window)

41. The public demo of the Maritime Single Window can be accessed on the following URL:

<https://imo-msw-public-test.azurewebsites.net/>

.

---

### Annex A Spreadsheet Templates

The Maritime Single Window uses an extended version of the JRCC eAPIS template for uploading bulky data.

## Crew- & Effects List (spreadsheet)

File No.	Last Name	First Name	Maritime Name	Nationality (ISO 3166-1 Alpha-3 Code)	Age (Y)	Date of Birth (DDMMYYYY)	Passport Number	Country of Issue (ISO 3166-1 Alpha-3 Code)	Date of Expiry (DDMMYYYY)	Port of Issuance (ISO 3166-1 Alpha-3 Code)	Port of Destination (ISO 3166-1 Alpha-3 Code)	Port of Call (ISO 3166-1 Alpha-3 Code)	Ship to calling	Place of work	Comments (to be filled from customs, police and visa or subject to jurisdiction as applicable)
1	THANG	JOHN		NOR	M	1964-04-27	AB2495	NOR	2022-04-27	NORHU	NOROO	NORHU	Captain	Haegeland	1 laptop, 2 mobile phones, 1 hand watch, 1 AED, 1 PDA w/ 2 pads, PE
2	WELLES	TOMMY		nor	M	1973-03-24	80113964	nor	2021-12-12	NORHU	NORVG	NOROO	Master	Roostak	1 laptop, 1 cell phone, 1 AED, silver watch, PE
3	ADITYA	AFD SHARMA		ind	M	1983-04-02	7F19890A	ind	2021-11-11	NORHU	NORVG	NOROO	Cabin Steward	SINGAPORE	1 laptop, 2 mobile phones, 1 AED, PE
4	ALBANO	ROBERTO	ita	ita	M	1983-06-25	2118HGFP	ita	2021-12-12	NORHU	NORVG	NOROO	Passenger Proxy	Manly	1 mobile phone, 2 wrist watch, 1 AED, PE
5	WINFIELD	THOMAS		nor	M	1987-07-02	023811584	nor	2021-12-12	NORHU	NORVG	NOROO	Engine Room & Te	Roostak	1 mobile phone, 1 AED, 1 laptop, PE
6	ANDER	KATHEN	DEU	F	1982-10-16	P2334484	DEU	2021-01-31	NORHU	NORVG	NOROO	Galley	Helsing	1 AED, 1 laptop, 2 USB, 1 wrist watch, 2 mobile phones, 2 tooth light	
7	ALAMEYAM	RUZ		ind	M	1981-10-07	02384484	ind	2021-12-12	NORHU	NORVG	NOROO	Utility Galley	Manly	1 laptop with charger, 1 wrist watch
8	ADITY	SHAR		ind	M	1983-10-11	023811583	ind	2021-11-11	NORHU	NORVG	NOROO	Cabin Steward	San Juan Isl	1 mobile phone, 1 digital camera, 1 USB flash drive, 1 wrist watch, PE
9	ALI	BEJANAB ADOD BN RAH		ind	M	1985-07-24	P2381151A	ind	2021-12-12	NORHU	NORVG	NOROO	Assistant Deck	PURBANGA	1 mobile phone, 1 AED, PE
10	ALPIDE	FEDRICO JR		nor	M	1977-10-22	0204998C	nor	2021-12-12	NORHU	NORVG	NOROO	Plumber	JMSTRT MTS O	1 mobile phone, 1 AED, 1 watch, PE

## Pax List (spreadsheet)

File No.	Last Name	First Name	Maritime Name	Nationality (ISO 3166-1 Alpha-3 Code)	Age (Y)	Date of Birth (DDMMYYYY)	Passport Number	Country of Issue (ISO 3166-1 Alpha-3 Code)	Date of Expiry (DDMMYYYY)	Port of Issuance (ISO 3166-1 Alpha-3 Code)	Port of Destination (ISO 3166-1 Alpha-3 Code)	Port of Call (ISO 3166-1 Alpha-3 Code)	Ship to calling	Place of work	Comments (to be filled from customs, police and visa or subject to jurisdiction as applicable)
1	THANG	JOHN		NOR	M	1964-04-27	AB2495	NOR	2022-04-27	NORHU	NOROO	NORHU	781234	Haegeland	Toward
2	ABDEL KADER	SAJJAH YKUME		ind	M	1975-12-12	0218HGFP	ind	2021-12-12	NORHU	NORVG	NOROO	11111	Layang	Toward
3	ADHANI	SANDRA		ind	M	1975-12-12	P2334484	ind	2021-12-12	NORHU	NORVG	NOROO	11111	Manly	Toward
4	ADENARI	PETRA		ind	M	1975-12-12	02384484	ind	2021-12-12	NORHU	NORVG	NOROO	11111	Helsing	Toward
5	ADENARI	SANDRA		ind	M	1975-12-12	0204998C	ind	2021-12-12	NORHU	NORVG	NOROO	11111	Manly	Toward
6	ADENARI	ANJA		ind	M	1975-12-12	P2381151A	ind	2021-12-12	NORHU	NORVG	NOROO	11111	Manly	Toward
7	ADENARI	NIHARIZ		ind	M	1975-12-12	0204998C	ind	2021-12-12	NORHU	NORVG	NOROO	11111	Roostak	Toward
8	ADENARI	HELGA		ind	M	1975-12-12	P2381151A	ind	2021-12-12	NORHU	NORVG	NOROO	11111	Roostak	Toward
9	ADENARI	BOONHANG		ind	M	1975-12-12	P2381151A	ind	2021-12-12	NORHU	NORVG	NOROO	11111	Manly	Toward

# Ship Stores List (spreadsheet)

IMO Maritime Single Window

**SHIP STORES DECLARATION**

IMO FAL Form 3

Name of article	Quantity	Location on board
Frozen meat	55	Provision Room, Band Store
Poultry	22	Provision Room, Band Store
Sausage	2000	Provision Room, Band Store
Bacon	30	Provision Room, Band Store
Fish	55	Provision Room, Band Store
Butter	22	Provision Room, Band Store
Margarine	2000	Provision Room, Band Store
Rice	30	Provision Room, Band Store
Vegetables	55	Provision Room, Band Store
Fruits	22	Provision Room, Band Store
Eggs	2000	Provision Room, Band Store
Bread	30	Provision Room, Band Store
Sugar	55	Provision Room, Band Store
Coffee	22	Provision Room, Band Store
Tea	2000	Provision Room, Band Store
Spirits	30	Banded Store
Dist Wine	30	Banded Store

Worksheet tabs: Crew List, Passenger List, Ship Stores Declaration

### Annex B Spreadsheet Upload diagram

Below is an outline of the process and logic of the upload function in Maritime Single Window.

## Spreadsheet – Upload and process logic

