# FIMS front-end code description

## Frameworks

The project is using TypeScript (https://www.typescriptlang.org) language and based on ReactJS (https://reactjs.org) and MobX (https://mobx.js.org) frameworks with Material-UI components (https://material-ui-next.com).
The map is based on Leaflet (https://leafletjs.com) framework.
The language support is based on i18next (https://www.i18next.com) framework.
For styling it uses PostCSS (https://postcss.org) transformation tool.

## Paths

*./src/* - sources
*./src/index.tsx* - index file
*./src/API.ts* - API calls functions
*./src/Utils.ts* - some common utility functions
*./src/containers/* - main view components
*./src/containers/App.ts* - index view component, everything starts from here
*./src/forms/* - forms components
*./src/forms/models/* - models for forms components
*./src/html/* - static files like fonts and images
*./src/stores/* - data stores for project data
*./src/styles/* - views and components styles
*./src/uicomponents/* - extensions for Material-UI components
*./build/* - build results folder
*./config/* - service files for building development and product versions
*./node_modules/* - JS library files, it will be created automatically when you run *npm install*
*./public/* - static public files folder, they will be copied automatically to *./build/* folder when you build the project
*./public/locales/* - translations files, en.json, vi.json
*./scripts/* - service files for building development and product versions
*./typings/* - typing files for TypeScript
*./tsconfig.json* - TypeScript config file
*./tslint.json* - TsLint config file
*./idea-codestyle-tslint.xml* - code style setting for IntelliJ IDEA IDE
./submobules/ external submodules, used in the project

# The code structure

## Top level components

The main html file is *./public/index.html*
The script workflow starts from *./src/index.tsx* and goes to *./src/containers/App.tsx*
All page routes are in *./src/containers/App.tsx*, so you can see almost the whole structure of the project there. *App.tsx* contains two sections; for whether the user is authorized and not.
For non-authorized user *./src/containers/auth/ContainerAuth.tsx* component is the container for all non-authorized forms.
For authorized users *./src/containers/global/ContainerGlobal.tsx* component is the container for all authorized forms. The header is in *AppHeader.tsx*.

## Localisation

Language files are static and you can find them in *./public/locales/* folder - *en.json* and *vi.json*.

## Map

All maps components are in *./scr/global/map/ folder.* The map initialization is in *Maps.tsx*.
The list of basemaps and layers is in /src/containers/global/map/MapLayers.ts in constructor(). It's just a simple array and it's very easy to change it. All other things (also the legend) the script will do automatically.
The keys of "layersData" array are the keys for translation file (map.* in en/vi.json), e.g. "forestFunctionMainClass: 'FRMS:Forest_function_main'" will show "FRMS:Forest_function_main" layer with "Forest function, main class" label.

## Login, registration

Login, Registration, Forgot password and Email confirmation forms are in *./scr/containers/auth/* folder.

## Forms

Enterprise management forms are in *./scr/containers/administrator/enterprises/* folder.
Accounts management forms are in *./scr/containers/administrator/accounts/* folder.
Factory add and edit forms are in *./scr/containers/enterprise/FactoryEditor.tsx*.
Raw material input form is in *./scr/containers/enterprise/MaterialInputAdd.tsx*.
Products output form is in *./scr/containers/enterprise/ProductionOutputAdd.tsx*.
Inspection reports upload forms are in *./scr/containers/inspectionReports/* folder.
Reports forms are in *./scr/containers/reports/* folder.
When you are creating a forms you can use ready-made components from *./scr/forms/* (if they need a validation) or raw components from Material-UI components framework (https://material-ui.com/demos/app-bar/).

### API

All API calls go through *./src/API.ts* library. API description can be found here: https://docs.google.com/document/d/1FWYI9u8eWEN-WVd7q7HMIyWJQlVOg2ltHRwB17LmbuE/edit

### Debug mode for development

Debug mode can be enabled/disabled in *./src/Utils.ts* by setting *Utils.DEBUG* variable to true/false.
The automatically generated code structure and descriptions by TypeDoc can be found here: https://drive.google.com/drive/u/3/folders/1hzbl3H54Fo1WuYs0JaYtd1TVuEOTLWr_ But it's very important to understand that this description is for the source code after some interpretations, so it's not the description of the pure sources.

## Stores

All user data is saved in *./src/stores/* components. Some interfaces are also here.
*Account.ts* - user account data store after he log in
*Browser.ts* - browser history data and url control center
*Config.ts* - config data that comes from the server
*DataFactory.ts* - some data that we need for user registration
*Enterprise.ts* - enterprise data interfaces and collections
*Factory.ts* - factories data store, interfaces and collections
*InspectionReport.ts* - inspection report interface
*ItemChooser.ts* - model class for choosing items in the list
*ItemCollection.ts* - item collections functions
*Local.ts* - browser localStorage functions
*Permissions.ts* - permissions for all application functions
*Store.ts* - the central place to coordinate all stores
*User.ts* - user profile data store

## Styles

All styles are in *./src/styles/* folder.
*_variables.scss* - global variables file