**Use cases**

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Login | |
| **Actors** | User | |
| **Parameters** | ID, Password | |
| **Preconditions** |  | |
| **Actions** | The user starts the system  The system presents a “Login Screen” to the user  The user inserts his ID and Password  The user presses the “Login” button  The system verifies the user’s input | |
| **Result** | The system grants permission to log in or raises the relevant error | |
| **Data** | | **Result** |
| The user is registered and inserts his ID and the correct password | | Pass |
| The user is not registered | | Fail |
| The user is already logged in | | Fail |
| The user inserts his ID and an incorrect password | | Fail |
| The user inserts incorrect ID and the correct password | | Fail |
| The user inserts ID that is shorter or longer than a valid ID number | | Fail |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Use Case Name** | | Register a new user | | | |
| **Actors** | | Project Manager / Contractor | | | |
| **Parameters** | | User ID | | | |
| **Preconditions** | | The project manager / contractor is logged in | | | |
| **Actions** | | The project manager / contractor is typing the user’s ID | | | |
| **Result** | | A new user is created | | | |
| **Data** | | | | **Result** | |
| The project manager / contractor types a valid ID | | | | Pass | |
| The project manager / contractor types a wrong ID | | | | Fail | |
| **Use Case Name** | Assign a role to a user | | | |
| **Actors** | Project Manager / Contractor | | | |
| **Parameters** | User ID | | | |
| **Preconditions** | The project manager / contractor is logged in | | | |
| **Actions** | The project manager / contractor is typing the user’s ID and assigning a new role to it | | | |
| **Result** | The user is assigned to a new role | | | |
| **Data** | | | **Result** | |
| The project manager / contractor types a valid ID and choosing a role | | | Pass | |
| The project manager / contractor types a wrong ID | | | Fail | |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Unassign a role to a user | |
| **Actors** | Project Manager / Contractor | |
| **Parameters** | User ID | |
| **Preconditions** | The project manager / contractor is logged in  The user has a role | |
| **Actions** | The project manager / contractor is typing the user’s ID and unassigning its role from it | |
| **Result** | The user is unassigned from its role | |
| **Data** | | **Result** |
| The project manager / contractor types a valid ID and choosing a role | | Pass |
| The project manager / contractor types a wrong ID | | Fail |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Create a new project | |
| **Actors** | Contractor | |
| **Parameters** | Project name | |
| **Preconditions** | The contractor is logged in | |
| **Actions** | The contractor clicks the “Create a new project” button and enters its name | |
| **Result** | A new project is created | |
| **Data** | | **Result** |
| The contractor clicks the “Create a new project” button and enters a valid name | | Pass |
| The contractor clicks the “Create a new project” button and enters a non-valid name | | Fail |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Create a project template | |
| **Actors** | Contractor | |
| **Parameters** | Project, template name | |
| **Preconditions** | The contractor is logged in, there is at least 1 project | |
| **Actions** | The contractor chooses a project  The contractor selects “Save as a template” | |
| **Result** | A new project template is created | |
| **Data** | | **Result** |
| The contractor clicks the “Save as template” button and enters a valid name | | Pass |
| The contractor clicks the “Save as template” button and enters a non-valid name | | Fail |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Create a new project from a pre-defined template | |
| **Actors** | Contractor | |
| **Parameters** | Project name, old template | |
| **Preconditions** | The contractor is logged in, the new project’s name is valid | |
| **Actions** | The contractor clicks the “Create a new project from template”  The contractor selects requested template and enters a new name | |
| **Result** | A new project is created | |
| **Data** | | **Result** |
| The contractor selects “Create a new project from template” but none exists | | Fail |
| The contractor clicks the “Create a new project from template” button and chooses a template | | Pass |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Edit a project | |
| **Actors** | Contractor | |
| **Parameters** |  | |
| **Preconditions** | The contractor is logged in | |
| **Actions** | The contractor chooses one of its projects  The contractor clicks the “Edit the project” button  The contractor edits the wanted details | |
| **Result** | The project is newly edited | |
| **Data** | | **Result** |
| The contractor clicks the “Edit the project” button and edits the wanted details with valid datum | | Pass |
| The contractor clicks the “Edit the project” button and edits the wanted details with non-valid datum | | Fail |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Add a new stage to a project | |
| **Actors** | Contractor | |
| **Parameters** | The contractor is logged in | |
| **Preconditions** |  | |
| **Actions** | The contractor clicks the “Add a new stage” button  The contractor writes the wanted details and adds the stage | |
| **Result** | A new stage is added to the project | |
| **Data** | | **Result** |
| The contractor clicks the “Add a new stage” button and writes the wanted details | | Pass |
| The contractor clicks the “Add a new stage” button and does not write the wanted details | | Fail |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Edit a stage in a project | |
| **Actors** | Contractor | |
| **Parameters** | The contractor is logged in | |
| **Preconditions** |  | |
| **Actions** | The contractor chooses a project  The contractor chooses stage  The contractor selects edit stage and edit with non-empty datum | |
| **Result** | The stage is newly edited | |
| **Data** | | **Result** |
| The contractor edits a stage with the wanted details | | Pass |
| The contractor edits a stage with empty data | | Fail |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Removes a stage from a project | |
| **Actors** | Contractor | |
| **Parameters** | Stage | |
| **Preconditions** | The contractor is logged in | |
| **Actions** | The contractor chooses a stage  The contractor clicks the “Remove the stage” button | |
| **Result** | The selected stage is removed from the project | |
| **Data** | | **Result** |
| The contractor clicks the “Remove the stage” button | | Pass |
| The stage doesn’t exist | | Fail |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Add a new mission to a stage | |
| **Actors** | Contractor | |
| **Parameters** | Mission details, stage | |
| **Preconditions** | The contractor is logged in | |
| **Actions** | The contractor enters to a stage  The contractor clicks the “Add a new mission” button  The contractor writes the wanted details and adds the mission | |
| **Result** | A new mission is added to the selected stage | |
| **Data** | | **Result** |
| The contractor clicks the “Add a new mission” button and writes the wanted details | | Pass |
| The contractor clicks the “Add a new mission” button and does not write the wanted details | | Fail |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Edit a mission in a stage | |
| **Actors** | Contractor | |
| **Parameters** | The contractor is logged in | |
| **Preconditions** |  | |
| **Actions** | The contractor chooses a project  The contractor chooses stage  The contractor chooses mission  The contractor selects edit mission and edit with non-empty datum | |
| **Result** | The mission is newly edited | |
| **Data** | | **Result** |
| The contractor edits a mission with the wanted details | | Pass |
| The contractor edits a mission with empty data | | Fail |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Removing a mission from a stage | |
| **Actors** | Contractor | |
| **Parameters** | The contractor is logged in | |
| **Preconditions** |  | |
| **Actions** | The contractor enters to a stage  The contractor chooses a mission  The contractor clicks the “Remove the mission” button | |
| **Result** | The selected mission is removed from the stage | |
| **Data** | | **Result** |
| The contractor clicks the “Remove the mission” button | | Pass |
| The mission doesn’t exist | | Fail |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Updating Mission Status | |
| **Actors** | Contractor, Project Manager, Work Manager | |
| **Parameters** | Project | |
| **Preconditions** | The actor is logged in | |
| **Actions** | The actor chooses a project  The actor chooses stage  The actor chooses a mission  The actor selects "Update mission Status"  The actor selects new status | |
| **Result** | The mission status will change accordingly | |
| **Data** | | **Result** |
| The actor has access to the project | | Pass |
| The actor doesn’t have access to the project | | Fail |
| Work Manager sets status from “invalid” to “Done” | | Fail |
| Project Manager/ Contractor sets status from “invalid” to “Done” | | Pass |

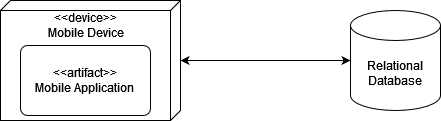
|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Viewing Project Information | |
| **Actors** | Contractor, Project Manager, Work Manager | |
| **Parameters** | Project | |
| **Preconditions** | The actor is logged in | |
| **Actions** | The actor selects a project | |
| **Result** | The actor receives all the project information | |
| **Data** | | **Result** |
| The actor has access to the project | | Pass |
| The actor does not have access to the project | | Fail |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Uploading proof to a mission | |
| **Actors** | Work Manager | |
| **Parameters** | proof file, project, mission | |
| **Preconditions** | The work manager is logged in and has access to the project | |
| **Actions** | The work manager selects a project  The work manager selects a stage  The work manager selects a mission  The work manager selects “upload proof”  The work manager selects and uploads the proof file from his device | |
| **Result** | The proof file is uploaded and saved for the specific mission | |
| **Data** | | **Result** |
| The work manager uploads a file with a decent size | | Pass |
| The work manager uploads a file with a very large size | | Fail |
| The work manager uploads an empty file | | Fail |

**System Architecture**

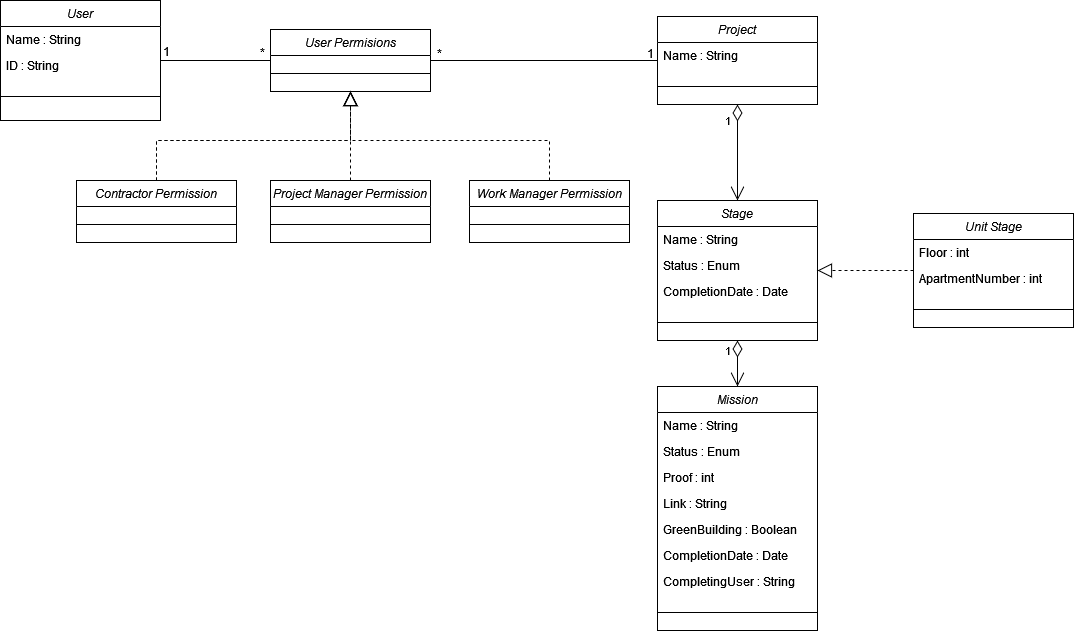
Mobile application: Written in Typescript and using React Native, the mobile application will run on the user’s phones and every query will require internet access. Data will not be saved locally.

Relational database: An SQL database located on a remote server, responsible for saving all the data object’s attributes.

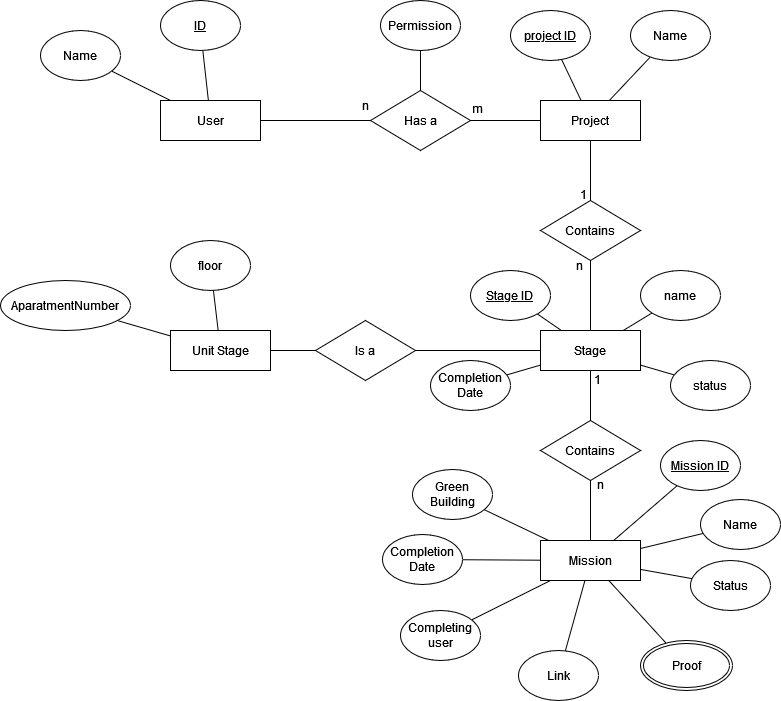


**Data Objects**

Class diagram representing our data domain:



Entity relationship model:



**User Interface Draft**

Our goal is to create an intuitive, easy to use, comfortable user interface.  
Note that all screens have a visible “Back” button and relevant screens have titles in the top, indicating the “path” the user has taken in the app. This will provide better “orientation” for the user while navigating the app.

**Startup Screen:**

**Graphical user interface, application

Description automatically generated**

The user inputs Username and Password by pressing on the blank box which opens the keyboard on their mobile device. The user types the required data with the keyboard.  
The user then presses the Login button. If the login failed the error message will be displayed in red text. If the user logged in successfully the application displays the next screen.

**Graphical user interface

Description automatically generated with low confidence**  
**All Projects Screen:**

The system displays only the relevant projects for the logged user, by showing buttons with the projects names. By clicking a button the system will display the next screen.

**Graphical user interface, application

Description automatically generatedProject Properties Screen:**

The system displays the project properties which the user can choose from. Each property will send the user to a different screen. “Building Stages” button will display “Build Stages Screen”, and “Building Faults” will display “Building Faults Screen”.

**Building Stages Screen:**

****

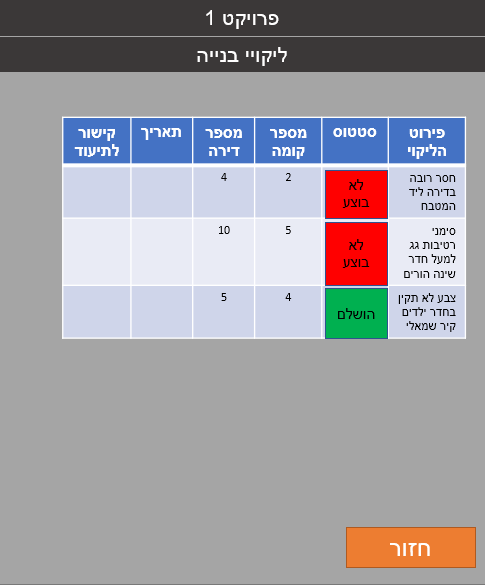
The system displays the list of stages in the chosen project. The project name can be seen at the top of the screen. Also a Back button is displayed which returns us the “All Projects Screen”  
The stages names are buttons, by clicking a stage button the system will display the next screen.

**Current Stage Screen:**

**Timeline

Description automatically generated**

The system displays the list of missions for the chosen stage in the chosen project. The system displays at the top of the screen the project name and the stage name under it. This will help the user to keep his “orientation” while navigating the system.  
Missions status can be changed by clicking of the status button for a specific mission, which opens a drop-down menu to select the new status from.

**Building Faults Screen:**

The system displays the building faults with all the relevant data as seen in the picture.

**Application Navigation Tree**

**Startup Screen**

**All Projects Screen**

**Projects Properties Screen**

**Building Faults Screen**

**Building Stages Screen**

**Current Stage Screen**

**Testing**

Our project will be developed using TDD method. That means tests for new features and requirements will be written before the code that implements them.  
Our tests will be composed of unit tests, integration tests and acceptance tests.  
We will be testing data reliability, system functionality, but we will not do any performance testing since the number of users and actions in the system is very narrow and high performance is not required.

Since usability is a top priority in our project, we will be giving a lot of weight to acceptance tests and customer tests, to receive continuous feedback and improve during development.  
Part of these tests are manual and will be done by the developers and the customer.

**Functional requirements testing:**

These tests are already written under every Use Case in the “ARD” document.  
To prevent duplication, the tests will not be written again, but you can find them in the ARD document under “Use Cases”

**Non-Functional requirements testing:**

The system should be able to support 50 users with up to 1 second reaction speed

We will use tools for load and performance testing to simulate 50 users sending actions in our application and measure the reaction time. There are a few tools that provide those capabilities and produce reports such as: Gatling, Apache Jmeter, React Native Performance Monitor.

The system should be compatible with Android and iOS

The app will be installed on smartphones from both operating systems. A sanity test, with a defined flow and some basic scenarios will be run on every OS. Those tests will be manual tests.

The system should support text in Hebrew (see “Dictionary” below)

This is a manual test. The tester will go through every screen and button in the application and compare it with the “Dictionary” found in the ARD document.

The system should support saving at least 500 previous projects

This will be part of the acceptance tests. An automated test that will create 500+ projects and verify their existence in the history.