תמונה שמכילה בחוץ, עץ, שמיים, צמח

התיאור נוצר באופן אוטומטי**Characterization document**

**Holon Institute of Technology mapping project**

**Project team:  
Tal Lilo, ID: 206361321**

**Dor Mizrahi, ID: 315429175**

**Nir Katz, ID: 206361883**

**Yamit Segev, ID: 206776486**

**Adir Yehuda Admon, ID:318999042**

**Instructor:  
Mr. Mark Israel**

**Mr. David Iolovich**

**תמונה שמכילה צילום מסך, עיצוב

התיאור נוצר באופן אוטומטי**

**Background:**  
The Holon Institute of Technology (HIT) is a large institute with many buildings and classrooms, and it can be difficult for students and visitors to find their way around. This is especially true for students and new visitors, who may not be familiar with the campus layout. The campus is advanced and has many sensors that needs to be controlled.  
The sensors will be part of our reference map.  
The guard will be represented as a point on the map  
And when we click on it, there will be information about the sensor that is ID, location and name.  
  
  
**The purpose:**To develop a flexible observation of the campus on different levels with the ability to add and remove and update objects easily. Another key functionality is product for sensors management as the light mood and sensors that provide information at the HIT and in a variety of places.

**Description of the current situation**There is a schematic mapping of a technological institute in a 3D environment. We used the QGIS.js plugin, which is three.js, which enables export for mapping by layers of the technological institute and transfers to a web display.

**Navigation in the system:**

-You can move around the buildings  
-You can zoom in and zoom out.

**Display in the system:**

-View of the buildings at the institute.  
3D view of the institute's area.-  
-Disappearing objects in the system.  
-Lowering transparency in the system.  
-Display of the classrooms inside the buildings.

**Information:**

-Local coordinates of the buildings in the area are exposed.  
-You can see the class numbers.  
-You can see the names of the classes such as the library and more.  
-You can see the layer of the selected building.

**The disadvantages:**

Long and cumbersome code and also unreadable.-  
-Difficulty in adding information or data, each object requires specific changes and adjustments.  
The code is very specific to a certain export that does not allow reusability.-  
-There is no modularity in the code.

-The code is written in pure js and no new technologies are used that enable more convenient and more advanced implementation techniques that enable faster development and a supportive community.  
-The plugin's documentation is not extensive enough and is more suitable for the user than the developer.  
-There is no documentation of the previous system we built, meaning there is no explanation of the code.

**The advantages:**

we get a web format of the environment in 3D very quickly with layers and data.-  
-There is already written code that works and it is always better to try and reuse than writing a new code.

**Stakeholders**  
 -HIT Institute of Technology: move to smart campus, accessibility for students and lecturers.

-Students: find their way around, navigate between buildings.

-Visitors: navigate themselves in the institute.

-Those interested in studying: reach information centers, open days.

-Staff people: orientate themselves, get to places.

-Academic Developers: manage sensors, their location, receive information.

**Functional requirements:**

|  |  |  |
| --- | --- | --- |
| **Description** | **The requirements** | **No**. |
| Registered user, admin user, viewer user. | There are three types of users. | **1** |
| The login and registration process for the system will be different according to different types of users, according to their  permissions. | There is a process of registration and connection to the system. | **2** |
| Permissions will be for adding, deleting and updating sensors, buildings and classes. | There will be permissions | **3** |
| Every registered user will receive permissions from the admin to what he is allowed to add / edit, delete and read. | Receiving permissions | **4** |

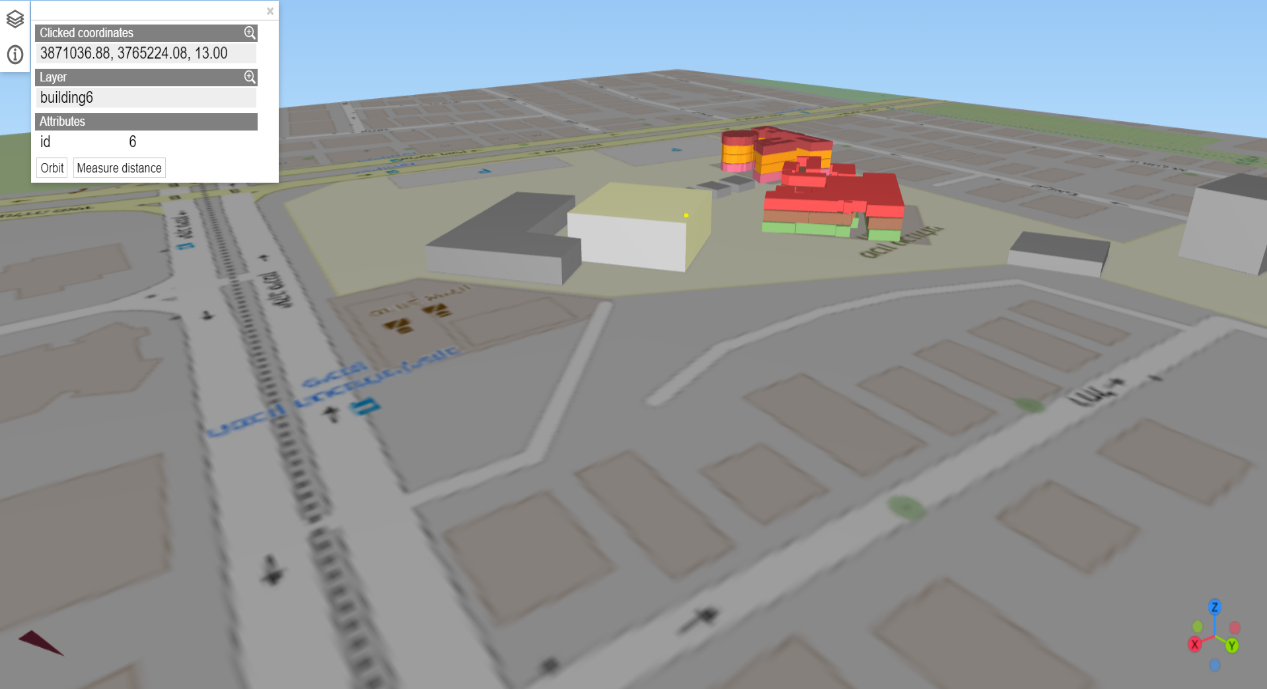
**Data Mapping**

We need to upload new photos through the new platform for our new improved database.

Every image should consist of properties demanded by the server.

Changing the available structure shall be executed by Edit Mode, enabling the registered user full access to entities that appear on the browser. The designated object will be sent to the front workspace, containing coordinates and azimuths. After uploading new data successfully to the front, new red points will appear on the screen.

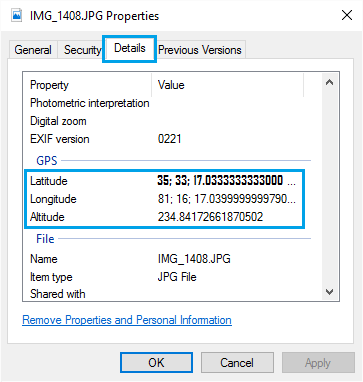
Note: Due to a lack of data, we can only see buildings with flat roofs. \*



When a registered user finishes editing or adding new data, the object is sent to edit mode, meaning that only the expected result can be transferred to the edit mode.

As a backup, any damaged or incomplete files won't reach the front.

Users will see only successfully uploaded files on the web page and object properties such as edges, vexes, coordinates, and azimuths. Leaflet, an open-source JavaScript Library is our tool for building web page mapping applications.



While adding a new image, a Windows Navigator should appear on the screen for choosing a document to upload.

In smartphones, third-party apps should solve the issue of lack of geo Data.

**Viewer Profile:**

|  |  |  |
| --- | --- | --- |
| **Description** | **The requirements** | **No.** |
| Accessible to perspective. | Looks at the structure from an aerial perspective | **1** |
| In order to see the classes inside the buildings. | Set the brightness. | **2** |
| In order to orientate in the institute. | Searching for classes | **3** |
| In order to view information  about the classes and the buildings such as: class number and building number | Receiving information | **4** |
| In order to get images of the college from different perspectives depending on the AZIMUTE and the GPS coordinates from which he will look | Receiving images | **5** |

**As a viewer**

**The login screen (clicking on the login button as a guest)**

**The app screen**

**Entering the name of the class and the building**

**Option to navigate between the buildings and classrooms - zoom**

**Option to click on objects and get information about them**

**Looking at the found class + getting information about it, and lowering the transparency of the structure**

**Registered user Part I**

|  |  |  |
| --- | --- | --- |
| **Description** | **The requirements** | **No.** |
| Adding objects that have permission such as sensors, classroom buildings and lanterns. | Adding objects | **1** |
| Deleting objects that have permission such as sensors, classroom buildings and lanterns. | Deleting objects | **2** |
| To move objects that I have permissions to, from one place to another. | Moving objects | **3** |
| to change object's values that I have permissions to, such as sensors I have access to. | Changing object's values | **4** |
| In order to view information about the classes and the buildings such as: class number and building number I have access to. | Receiving information | **5** |
| It is not possible to put classes into the structures whose size exceeds the structure, if the user tries he will get an error. | User Warning | **6** |

**Registered user Part II**

|  |  |  |
| --- | --- | --- |
| **Description** | **The requirements** | **No.** |
| As soon as the class is added to a certain structure, it becomes part of the structure even when the structure is moved. | Classes's characteristics | **7** |
| Accessible to perspective. | Looks at the structure from an aerial perspective | **8** |
| In order to see the classes inside the buildings. | Set the brightness. | **9** |
| In order to orientate in the institute. | Searching for classes | **10** |

**Registered user**

**Login as a registered user**

**Login screen**

**Registration screen**

**Yes**

**No**

?Registered

**The app screen**

**Adding  
/changing objects according to permissions**

**Clicking on objects + receiving related allowed information according to permissions**

**Entering the name of the class and the building**

**Option to navigate between the houses and classrooms - zoom in/out**

**Account details screen**

**Looking at the found class + getting information about it, and lowering the transparency of the structure**

**Selecting the position adding an object**

**Selecting the location to delete or change values of an object**

**Change account details**

**User Administrator Part I:**

|  |  |  |
| --- | --- | --- |
| **Description** | **The requirements** | **No.** |
| The admin will allow permissions for different types of users. | allowing Permissions. | **1** |
| - | The admin can add permissions. | **2** |
| - | The admin can delete permissions. | **3** |
| - | The admin can delete or block users. | **4** |
| - | The admin can unblock users. | **5** |
| Accessible to perspective. | Looks at the structure from an aerial perspective | **6** |
| In order to see the classes inside the buildings. | Set the brightness. | **7** |
| In order to orientate in the institute. | Searching for classes | **8** |
| Adding objects that have permission such as sensors, classroom buildings and lanterns. | Adding objects | **9** |

**User Administrator Part II:**

|  |  |  |
| --- | --- | --- |
| **Description** | **The requirements** | **No.** |
| Deleting objects that have permission such as sensors, classroom buildings and lanterns. | Deleting objects | **10** |
| To move objects that I have permissions to, from one place to another. | Moving objects | **11** |
| to change object's values that I have permissions to, such as sensors I have access to. | Changing object's values | **12** |
| In order to view information about the classes and the buildings such as: class number and building number I have access to. | Receiving information | **13** |
| It is not possible to put classes into the structures whose size exceeds the structure, if the user tries he will get an error. | User Warning | **14** |
| As soon as the class is added to a certain structure, it becomes part of the structure even when the structure is moved. | Classes's characteristics | **15** |

**As a admin:**

**Login as a registered user**

**Login screen**

**The app screen**

**Adding/  
changing objects according to permissions**

**Clicking on objects + receiving related allowed information about users**

**Option to navigate between the buildings and zoom classrooms - zoom in/out**

**Account details screen**

**Entering the name of the class and the building**

**The permissions screen**

**Change account details**

**Looking at the found class + getting information about it, and lowering the transparency of the structure**

**Selecting the position adding an object**

**Selecting the location to delete or change values of an object**

**Selection of registered users**

**Change permissions**

**Users in the system:  
  
Registered user**

|  |  |  |
| --- | --- | --- |
| **Description** | **Feature** | **No.** |
| Type-String | User name | **1** |
| Type-String (Primary key) | Mail | **2** |
| Type-geo-json | Location | **3** |
| Type-String | Encrypted password | **4** |
| Type-array of string | Permission types | **6** |
| Type-array of schema object id | Created objects | **7** |
| Type boolean | User blocked or not | **8** |
| String | picture | **9** |

**Administrator user**

|  |  |  |
| --- | --- | --- |
| **Description** | **Feature** | **No.** |
| Type-String | User name | **1** |
| Type-String(Primary key) | Mail | **2** |
| Type-geo-json | Location | **3** |
| Type-String | Encrypted password | **4** |
| Type-array of schema object id | Created objects |  |
| String | picture |  |

**:The objects in the system**

**Sensors**

|  |  |  |
| --- | --- | --- |
| **Description** | **Feature** | **No.** |
| Type-geo-json | Location | **1** |
| Type-int(auto- incremental) (Primary key) | ID | **2** |
| Type-String | Name | **3** |

**Flashlights**

|  |  |  |
| --- | --- | --- |
| **Description** | **Feature** | **No.** |
| Type-geo-json | Location | **1** |
| Type-String | Kind | **2** |
| Type-float | Size | **3** |
| Type-int | Lighting intensity | **4** |
| Type-int(auto- incremental) (Primary key) | ID | **5** |

**Buildings**

|  |  |  |
| --- | --- | --- |
| **Description** | **Feature** | **No.** |
| Type-geo-json | Location | **1** |
| Type-int | Building number | **2** |
| Type-float | Size | **3** |
| Type-int | Classrooms | **4** |
| Type-int | Number of floors | **5** |
| Type-int(auto- incremental) (Primary key) | ID | **6** |

**Classes**

|  |  |  |
| --- | --- | --- |
| **Description** | **Feature** | **No.** |
| Type-geo-json | Location | **1** |
| Type-int | Building number | **2** |
| Type-int | Class number | **3** |
| Type-float | Class size | **4** |
| Type-int | Number of floors | **5** |
| Type-array of schema object id | Sensors | **6** |
| Type-int(auto- incremental) (Primary key) | ID | **7** |

**ERD**  
https://github.com/DorMizrahii/GIS/blob/main/Gis%20erd.jpg

**Non-functional requirements PART I:**

|  |  |  |
| --- | --- | --- |
| **Description** | **The requirements** | **No.** |
| The system will support the English and Hebrew languages. | Usability | **1** |
| The system will be available 24/7. | Reliability | **2** |
| The system will work on the website and will be available in Chrome, Edge, Safari and Firefox and will also be responsive | Availability | **3** |
| Presentation for tablets and smartphones. | Usability | **4** |
| The system will return a response to the user, while waiting no longer than 3 seconds. | Performance | **5** |
| The system will be able to serve about 5,000 customers at the same time. | Scalability | **6** |

**Non-functional requirements PART II:**

|  |  |  |
| --- | --- | --- |
| **Description** | **The requirements** | **No.** |
| The system will provide reliable information thanks to the sensors, lights and buildings. | Reliability | **7** |
| The system will work on the https protocol in order to be secure, and provide protection to the customer when browsing the site. | Security | **8** |
| The system will use PostgreSQL and back up its information in the cloud. | Space | 9 |
| If the user is logged in for more than fifteen minutes without activity, the system will take him out, in order to reduce the load on the servers - performance improvement. | Performance | **10** |
| The application will be a WEB application and for the cell phone it will be available the app store. | Accessible to other platform | **11** |
| The system is suitable for Linux, Windows, IOS, Android | Suitable for common operating systems | **12** |

**React vs Angular- Part I**

|  |  |  |
| --- | --- | --- |
| **The issue** | **React** | **Angular** |
| 1) Component-Based Architecture: | React is known for its component-based architecture. It allows to break down user interface into reusable components, which can be helpful when dealing with complex 3D GIS elements. Each component can manage its own state, making it easier to manage and update different parts of the 3D visualization. | Angular also offers a component-based architecture, where components encapsulate different parts of application. Angular's approach includes more built-in features, like two-way data binding, which might simplify some aspects of managing complex data interactions in a 3D GIS project. |
| 2) Integration with 3D Libraries | React's flexibility makes it suitable for integrating with various third-party libraries and tools. Libraries like Three.js, which is commonly used for creating 3D visualizations, can be seamlessly integrated into React applications. React's Virtual DOM can also help optimize rendering performance in dynamic 3D scenes. | Angular provides a robust ecosystem, including the ability to encapsulate third-party libraries in Angular components. While Angular might have a steeper learning curve due to its more opinionated structure, it can provide more guidance on integrating and managing complex libraries, which can be advantageous in a 3D GIS context. |
| 3) State Management | React offers a variety of state management solutions, with Redux being a popular choice. Managing the state of a 3D GIS application can be challenging due to the dynamic nature of data and interactions. Redux, along with React's unidirectional data flow, can help maintain a clear data flow and aid in debugging. | Angular includes its own state management system known as NgRx, which is heavily inspired by Redux. It provides tools for managing state and side effects in a predictable manner. This can be advantageous in a 3D GIS project where maintaining a consistent state is crucial. |

**React vs Angular- Part I**

|  |  |  |
| --- | --- | --- |
| **The issue** | **React** | **Angular** |
| 4) Performance | React's Virtual DOM and efficient rendering updates can contribute to better performance, especially when dealing with frequent changes in the 3D visualization. This can help ensure a smooth experience even with complex and dynamic 3D scenes. | Angular's change detection mechanism can sometimes be less performant compared to React's Virtual DOM, especially in scenarios with high-frequency updates. However, Angular's Ahead-of-Time (AOT) compilation can lead to faster initial loading times. |
| 5) Learning Curve | React's learning curve is generally considered to be more gradual, making it easier for developers new to the framework to get started. It's simple and focused API can be advantageous in a project where the development team might have varying levels of expertise. | Angular has a steeper learning curve due to its comprehensive ecosystem and complex concepts. However, its opinionated nature can provide a clear structure for development, which might be beneficial when working on a large-scale 3D GIS project with multiple developers. |

In summary, React's flexibility, performance optimization through Virtual DOM, component-based architecture, and strong ecosystem make it an excellent choice for developing a 3D GIS project. Its ability to handle complex data interactions, while also catering to developers with varying levels of experience, positions it as a suitable framework for our project's needs. Therefore we will choose React.

**DataBase:**

**PostgreSQL**

PostgreSQL is a relational database with support for advanced features like joins, transactions, and constraints. PostGIS is an extension for PostgreSQL that adds spatial and geographic capabilities, making it well-suited for GIS applications.  
PostGIS provides powerful spatial functions, indexing, and querying capabilities for spatial data. It's designed to handle complex geospatial data types, operations, and relationships.  
PostgreSQL enforces data integrity through its support for ACID transactions, which ensures data consistency and reliability.  
PostgreSQL can scale both vertically (by adding resources to a single server) and horizontally (by using replication and partitioning). However, scaling can be more complex than with NoSQL databases like MongoDB.  
PostgreSQL uses SQL, which is great for complex queries, reporting, and analytics.  
PostgreSQL has a mature and active community, along with various extensions beyond PostGIS that can enhance its functionality.

The project involves complex spatial relationships, querying, and transactions.

PostgreSQL's community has been around longer and has a wide range of extensions and community and It can combine with instance and dynamics.

Development environments in react:  
Recommended environment would be vs code for the following reasons:

* It is in the top of 7 IDEs in react-native.
* There are many extensions that support libraries like three.js and many react libraries.
* There is a quick option to develop a web application.
* It is convenient to use.
* There is a lot of information accessible on the Internet
* It's free.
* The VS Code environment allows the use of the prettier tool that helps with code corrections and enables work in an efficient environment that will advance the development process.
* A quick environment that enables the development of a 3D mapping system that requires the use of graph files.
* The VS Code environment is available in all operating systems on the market and enables consistency in the development process and is an advantage for a team that has team members with different operating systems.



Mapping Geolocation with Leaflet

Leaflet.js is an open-source library using which we can deploy simple, interactive, lightweight web maps.

Notes:

- Geolocation is crucial for Data mapping.

- Geolocation data is given by using GPS while using the camera.

- Geolocation can be added manually when a connection is unavailable.

Questions/Notes for the guide:

- How can I obtain the azimuth while taking a picture?

In smartphones, third-party apps should solve the issue of lack of geo Data. -

**Estimated requirements**

**Time frames** :

the estimated time is half a year to a year.

**Manpower:**

we estimate that six people are needed to work on the system

development:

2 for front-end,

2 for backend ,

1 for testing and integration,

1 for UI UX.

**Resources:**

data storage service,

design software programs like figma,

project Management Tools like git,

optional:zoom,

using 3D GIS third party library's.

frontend Framework

backend Framework

**Estimated costs:**

**optional:** 3D GIS libaray's for example arc gis – 75$ a month

cesiumcommercial plan 150$ a month

**data storage service –** 3.5$ a month –

Amazon S3 (Standard Storage):

Let's assume we are storing 100 GB of data and have relatively low data transfer out (e.g., 10 GB):

Storage: 100 GB \* $0.026 (average) = $2.60

Data Transfer Out: 10 GB \* $0.09 (average) = $0.90

Estimated Monthly Total: $2.60 + $0.90 = $3.50

**System screens**

|  |  |  |
| --- | --- | --- |
| **Access** | **Screen's Name** | **No.** |
| Admin, registered user | Sign in | **1** |
| Registered user | sign up | **2** |
| registered user and admin | Account details update screen | **3** |
| Guest, registered user, admin | App screen | **4** |
| Admin | Permissions screen | **5** |
| registered user | Object management screen | **6** |
| admin guest and registered user | Object information screen | **7** |