**Vision and Scope Document**

**for ITA-Tools**

**Prepared by developer’s team**

**1** **System vision 3**

[**1.1** **Vision Statement**](#_tyjcwt) **3**

**1.2** [**Major Features**](#_tyjcwt) **3**

**1.3** [**Terms, abbreviation and definitions**](#_tyjcwt) **4**

**1.4** **Background 4**

**1.5** **Document Purpose 4**

[**2** **Scope and Limitations**](#_3dy6vkm) **5**

**2.1** **Scope of Initial Releases 5**

**2.2** **Scope of Subsequent Releases 5**

**2.3** **Limitations and Exclusions 5**

[**3**](#_3rdcrjn) [**Business Context**](#_3rdcrjn) **7**

[**3**](#_35nkun2)**.1** [**Project Priorities**](#_35nkun2) **7**

**3.2 Operating Environment 7**

**4 Project dependencies 8**

**5 Functional and Non-functional Requirements 9**

**5.1 Functional Requirements 9**

**5.2 Non-functional Requirements 9**

# System vision

# 1.1 Vision Statement

ITA-tools is a website where useful tools are stored. Tools may be used in the learning process, and in normal mode. Designed for students of IT Academy and people, forced to work in closely interacting teams. Using this site, user will have a convenient organizer with options, which take advantages in things like: structuring plans, keep track of his schedule, simplify the distribution of tasks between team members, messaging in team. ITA-tools can provide online messaging between users.

**1.2 Major Features**

The major features include:

* Student's personal account

1. Edit username, password
2. Log out
3. Sign in
4. Sign up

* Random selection of data from the provided list, with the possibility of presetting.

1. Add data
2. Select many from many
3. Select one from many
4. Generating unique results
5. Grouping data by specific options
6. Rerandom some parts of result

* Student tasks list

1. Create task
2. Edit task
3. Delete task
4. Sort tasks

* Chat for users

1. Send message
2. Receive message

* Site search functionality

1. Search on page
2. Search on site
3. Show results

## 1.3 Terms, abbreviation and definitions

|  |  |
| --- | --- |
| UML | Unified modeling language is a general-purpose, developmental, modeling language in the field of software engineering, that is intended to provide a standard way to visualize the design of a system |
| JS | JavaScript |
| CRUD | Create, retrieve, update, delete operations |
| REM | Role education model |
| React | Library for building interface |
| Redux | Pattern for managing application state |

## Background

SoftServe ITA employees currently devote extra time to organizing training.

Until the last few years, SoftServe ITA completely dispensed with applications to improve the learning system.

The driving factor behind the development of this software project is that there is nothing as in a market that can maintain a training system using tools developed by students of the academy for students.

**1.5 Document purpose**

The main goal of writing this document is to define high-level needs and features of the “ITA Tools”. It shows why the product should be developed and emphasizes the primary needs of clients and target users. Common details about work principles of the “ITA Tools” are described in the attachment.

**2. Scope and Limitations**

Given the vision, it can be understood, that all tools cannot be realized at once. In the sections that follow, the scope of this project will be defined in terms of major features, that will be implemented and those, that will not.

The main goal of this development cycle is to realize those features, that can satisfy our immediate needs. Our project will be enhanced in the future by adding new features.

**2.1 Scope of Initial Release**

The major features that will be implemented in our project are:

1. Randomizer tool for different purposes   
2. Account authorization(sign in/sign up)  
3. REM feature for distributing roles of education process  
4. Schedule feature for displaying days and time when different groups take a class.

**2.2 Scope of Subsequent Releases**

The main feature of subsequent development cycles is the addition some other features.

List of this features:

1. Todo list tool that shows tasks to be made
2. Photo gallery
3. “Chicken house” tool that helps to decide who of the group will be responsible for a particular task.
4. Chat for communication with other users
5. Data visualization tool that displays different graphs and chats.

**2.3Limitations and Exclusions**

There can be a lot of other tools implemented in our project to improve its functionality. This section lists desirable features that are outside of the scope of this project, but are available for addition at a later time.

Our project won`t have these features:

1. Possibility to change style of the web page
2. Quizzes on different topics to improve user’s skills
3. Archive that stores information about all groups that have completed

**3. Business Context**

This section summarizes some of the business issues around the project

**3.1 Project Priorities**

* Schedule
* Randomizer
* Photo Gallery
* Chat
* Search
* Authorization system

**3.2 Operating Environment**

This section will describe the environment in which the system will be used and define the major availability, reliability, performance, and integrity requirements.

The important operating environment information is:

1. The application will work on any operating system. The server will be any.
2. There is currently no database. Redux used to store state.
3. All users will be within the same time zone as the one in which the server is located.
4. The application is available at any time.
5. The data will be generated by clients and stored client-side

# 4. Project dependencies

Application will have next dependencies:

ReactJS + Redux + React Router +Firebase

**ReactJS**

React (also known as React.js or ReactJS) is a JavaScript library for building user interfaces. It is maintained by Facebook and a community of individual developers and companies

React can be used as a base in the development of single-page or mobile applications, as it is optimal for fetching rapidly changing data that needs to be recorded. However, fetching data is only the beginning of what happens on a web page, which is why complex React applications usually require the use of additional libraries for state management, routing, and interaction with an API: Redux, React Router and axios[ are examples of such libraries.

**Redux**

Redux is an open-source JavaScript libraryfor managing application state. It is most commonly used with libraries such as Reactor Angularfor building user interfaces. Similar to (and inspired by) Facebook's Flux architecture, it was created by Dan Abramov and Andrew Clark.

Reduxis a small library with a simple, limited API designed to be a predictable container for application state. It operates in a similar fashion to a reducing function, a functional programming concept.

**React router**

React router is a routing library built on top of the react which is used to create the routing in react apps.

**Firebase**

Firebase is a mobile and web application development platform developed by Firebase, Inc. in 2011, then acquired by Google in 2014.[[5]](https://en.wikipedia.org/wiki/Firebase#cite_note-5) As of October 2018, the Firebase platform has 18 products,[[6]](https://en.wikipedia.org/wiki/Firebase#cite_note-6) which are used by 1.5 million apps.

# 5. Functional and Non-functional Requirements

**5.1 Functional Requirements**

* HTML5 (BEM)
* CSS3
* TypeScript
* React (Hooks, Router)
* Redux
* NPM / YARN
* Webpack
* Babel
* Linter
* Prettier

**5.2 Non-functional Requirements**

Need to implement:

* Schedule
* Randomizer
* Photo Gallery
* Chat
* Search
* Authorization system

The important information is:

1. go to content should be 2 - 3 clicks;
2. The application will work on any operating system. The server will be any;
3. There is currently no database. Redux used to store state;
4. All users will be within the same time zone as the one in which the server is located;
5. The application is available at any time.