Bilkent University
Department of Computer Engineering

Senior Design Project
T2329
Block Script

Analysis and Requirement Report

21901637, Mehmet Kaan Örnek, kaan.ornek@ug.bilkent.edu.tr

21803341, Göksu Şenerkek, @ug.bilkent.edu.tr

21903613, Khashayar Amini, @ug.bilkent.edu.tr

21702990, Muhammed İkbal Doğan, ikbal.dogan@ug.bilkent.edu.tr

Supervisor: Can Alkan

Course Instructors: Atakan Erdem, Mert Bıçakçı

Dec 1, 2023

Contents

1 Introduction

In today's technology-driven world, the importance of software development is recognized by everyone. Technology has been one of the most promising sectors for many years, therefore, there is a tendency among people towards learning programming. People, who have been working in other sectors, quit their job to learn programming and find a job in this sector. The computer science departments of the universities are one of the most popular choices among the most successful students in the high schools. In addition to individual tendency, there is a great effort to embed programming as a core skill to society, especially to children. For example, in Turkey, private high schools have started to offer programming courses [1]. Many countries, like the United Kingdom, India and South Korea, integrated programming into their national curriculum of either primary schools or high schools [2, 3].

However, teaching programming to children is a challenge by itself considering the abstract aspect of programming languages and the poor attention span of children. Therefore, the need for additional tools, that eases teaching how to program to kids by providing representational units against abstractness and gamification or appealing user interface against poor attention span, has emerged. It is observed that such a tool for web programming is absent, and this project aims to provide a tool for this purpose.

1.1 Description

Block Script will be a visual programming language designed for education purposes. Block Script will be used to make the process of learning web development easy and fun for children using blocks that encapsulates JavaScript, HTML and CSS codes. Using our design tool, users will be able to create projects and web pages within the projects. The needed blocks will be seen in the screen to create the desired pages. Once the user finishes the development, they will be able to export the underlying code that they have generated using blocks. Block Script will prepare the corresponding Javascript, HTML and CSS files, and deliver the files to the user. Therefore, Block Script will provide a comprehensive and educational web programming language by giving the user the ability to design both the user interface and client-side logic of a website without the need to write any line of code.

2 Proposed System

2.1 Overview

Block Script will consist of two different components which are a design tool and a compiler. The design tool will be the interface that users will be integrating with. Using the design tool, users will be able to create projects and web pages within the projects by using and combining blocks. The needed blocks will be seen in the screen to create the desired pages. Once the user finishes the development, they will be able to export the underlying code that they have generated using blocks. To provide that code, we will have our compiler component. Compiler will generate the code and prepare the corresponding Javascript, HTML and CSS files which can be run as a website.

There will be mutual interaction between the design tool and the compiler. When a user wants to export their project, the information about which blocks are used and how they are combined with each other on which pages will be transmitted to the compiler component. The compiler will process this information and prepare the corresponding files. Once these files are prepared, it will transmit the files to the design tool, so that the users can download their resulting codes.

The only user type will be the users who are developing using the design tool and the compiler. Registration to the system will be optional, however, the projects of the registered users will be stored in the cloud and will be available. Therefore, registered users will be able to access their projects from different devices they log in. Users who aren't registered will be able to only local projects. Likewise, registered users will be able to reach their statistics about their account indicating the number of the projects and pages they created.

2.2 Functional Requirements

- Users should be able to sign up to the system by specifying username, password and email address.
- Users should be able to access the design tool by either logging in to a registered user or without logging in.
- Registered users should be able modify their username, password and email with verification through email.

Design Tool

- Design tool must allow users to create and delete a project.
- Design tool must allow users to see their previously created local projects.
- If the user is registered, the design tool must allow users to see their previously created projects that are in the cloud.
- Design tool must allow users to create and delete pages in a project.
- Design tool must allow users to assign a name to their projects and pages while creating.
- Design tool must allow users to change the name of an existing project or page.
- Design tool must show the code blocks available.
- Design tool must allow users to put and remove code blocks to the created pages.
- Design tool must allow users to connect blocks linearly.
- Design tool must allow users to put blocks in blocks.
- Design tool must allow users to preview the page they created on the same screen with blocks.
- Design tool must allow users to save changes in pages and projects.
- Design tool must allow registered users to save projects to the cloud.
- Design tool must allow users to export the actual code that they created using blocks.
- Design tool must allow registered users to see their achievements and statistics indicating the number of the projects and pages they created.
- The user interface of the design tool should be easy to use.
- The user interface should be engaging and fun to use.
- Tools provided by Block Script should be intuitive to use.
- All blocks needed to create a functional website should be accessible to the user.

- Compatible blocks should have compatibility in their visual design.
- Incompatible blocks should have incompatibility in their visual design.
- Design tool must allow users to access the settings screen to handle login information or theme.

Compiler

- The compiler should create code which is readable.
- The compiler should export the generated code as JavaScript, HTML, CSS files.
- The compiler should give an error if there exists a problem in the design.
- The compiler should be fast and reliable.

2.3 Non-functional Requirements

2.3.1 Usability

- The design tool will be used mostly by children, therefore, use of the tool shouldn't be complicated, instead, it must be easy to understand and use. The texts on the product, like menu or help page, must be prepared in consideration of children.
- To increase usability against poor attention span of children, the use of tools must be fun and the interface must be appealing and attractive.

2.3.2 Reliability

- The design tool should be reliable while saving and loading designs to and from disk. Any issue during this process will cause loss of user progress.
- The compiler should be reliable while compiling each design and generate correct and accurate output.

2.3.3 Performance

- The designer user interface should be responsive and have high performance to provide a better user experience.

- The part of the product that will use the most computation time is the compiler. Therefore, the algorithms in the compiler must be optimized carefully to generate the output files in a short amount of time.

2.3.4 Supportability

- The design tool is supportable by Windows, Linux and web. It will be builded from the same codebase for any type of system.
- In terms of program life cycle, users' design state will be saved locally. States will be reachable when the user opens it next time.
- In terms of any exception, software should respond to those exceptions.

2.3.5 Extensibility

- The blocks in the design tool will be predetermined and listed in the design tool. However, as technology improves, the need for different kinds of blocks might emerge. Therefore, extensibility must be considered in the architecture.

2.3.6 Scalability

- The design tool and compiler should work cooperatively independent from numbers of users. As the user's computer has enough resources, it should handle the design and compilation process.

2.3.7 Security

- Block Script will keep user’ login information for functional purposes, therefore, this information should be kept secure.
- The projects that are saved to the cloud should be kept secure as well.

2.4 Pseudo Requirements

- Implementation of the design tool will be done through Electron.js.
- Implementation of the compiler will be done with a high performance programming language like C/C++ or Golang.
- Version control will be done through Git.
- Cloud services will be provided by Firebase.
- Authentication service will be provided by Firebase.

2.5 System Models

2.5.1 Scenarios

Scenario 1: Signing Up

Actor: User

Flow of Events:

1. User opens Block Script.
2. User fills the fields that correspond to username, password and email address.
3. User gets an authentication email if the username and email address don't belong to an already registered user and the password and email address are valid.
4. User verifies their account through the email.

Scenario 2: Modifying Login Information

Actor: Registered User

Flow of Events:

1. User opens Block Script.
2. User logs in by filling the fields for username and password.
3. User accesses the design tool after logging in.
4. User clicks the “Settings” button in the design tool.
5. User clicks the “Manage your Block Script account” button in the settings screen.
6. User modifies the fields for username, password or email address, and clicks “Save”
7. User enters their current password to validate this action.

Scenario 3: Creating a Basic Login Page

Actor: User that is in the Design Tool and created a empty page

Flow of Events:

1. User drags and drops the “<html>” block which is available in the list of blocks to the editing screen.

2. User drags and drops the “<head>” block inside the “<html>” block.
3. User drags and drops the “<title>” block inside the “<head>” block and writes “Basic Login Page”.
4. User drags and drops the ”<body>” block inside the “<html>” block and next to the “<head>” block. Then, the user puts a “ ” block inside the “<body>” block. Likewise, the user puts a “<form>” block inside the “ ” block.
5. User puts a “<input>” block inside the “<form>” block and sets its type attribute to text.
6. User puts another “<input>” block next to the previous “<input>” block and sets its type attribute to password.
7. User puts a “<button>” block next to the “<input>” blocks and sets its text attribute to “Login”.
8. User clicks “Save”.

Scenario 4: Creating a Project with Two Pages and Exporting the Project

Actor: User that is in the Design Tool and logged in

Flow of Events:

1. User clicks the “Create a Project” button to create an empty project.
2. User enters “My Social Media” as the name of the project to the popped up screen.
3. User clicks the “Add a Page” button to add an empty page to the project.
4. User enters “Login” as the name of the page to the popped up screen.
5. User does events 3 and 4 again to create the second page and creates another page named “Home”.
6. User selects pages one by one, and develops a login and a home page through a similar approach explained in scenario 2.

7. User clicks the “Export” button and chooses a location in the device they are using. The corresponding HTML, CSS and Javascript files will automatically be downloaded to that location.

Scenario 5: Accessing Projects from Another Device

Actor: User

Flow of Events:

1. User opens Block Script and creates a project and develops several pages.
2. User opens Block Script on a different device that isn't the previous one.
3. User clicks the “Continue without logging in” button.
4. User realizes the projects aren’t visible because the device doesn’t have the projects previously created.
5. User clicks the “Settings” button, and fills the required fields after clicking the “Login” button in the settings page. Likewise, the user may restart Block Script to get to the login screen.
6. After logging in, the user sees the projects that were previously stored in the cloud by the system.

2.5.2 Use-Case Model

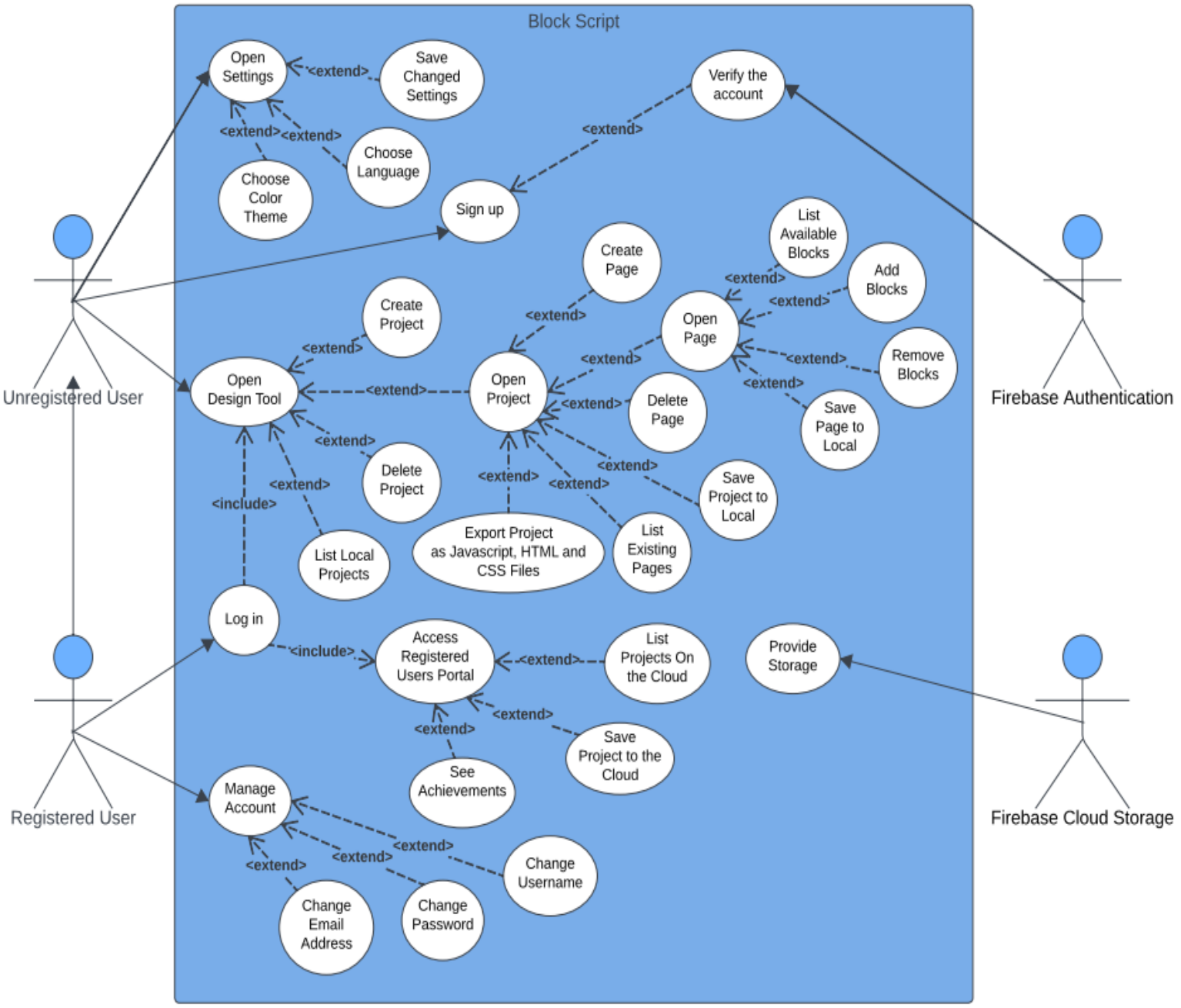

Figure 1: Use Case Diagram

2.5.3 Dynamic Models

2.5.3.1 Activity Diagrams

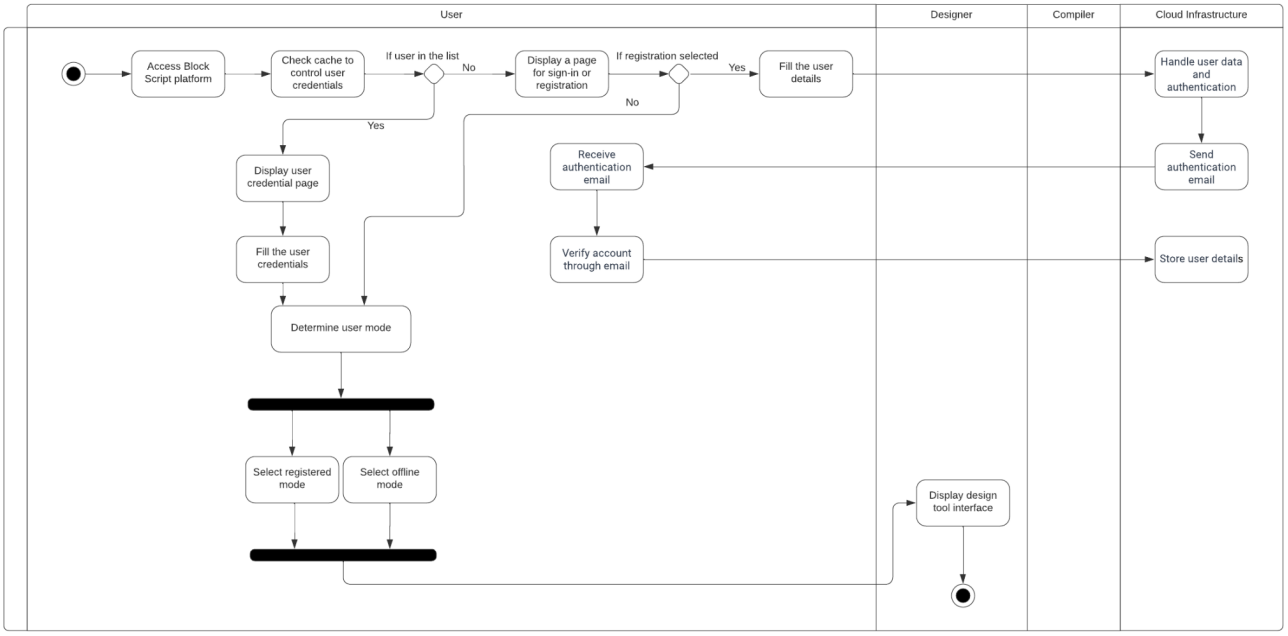

Figure 2: Activity Diagram of the User Registration and Login Process (For original resolution, see activity diagram1.png .)

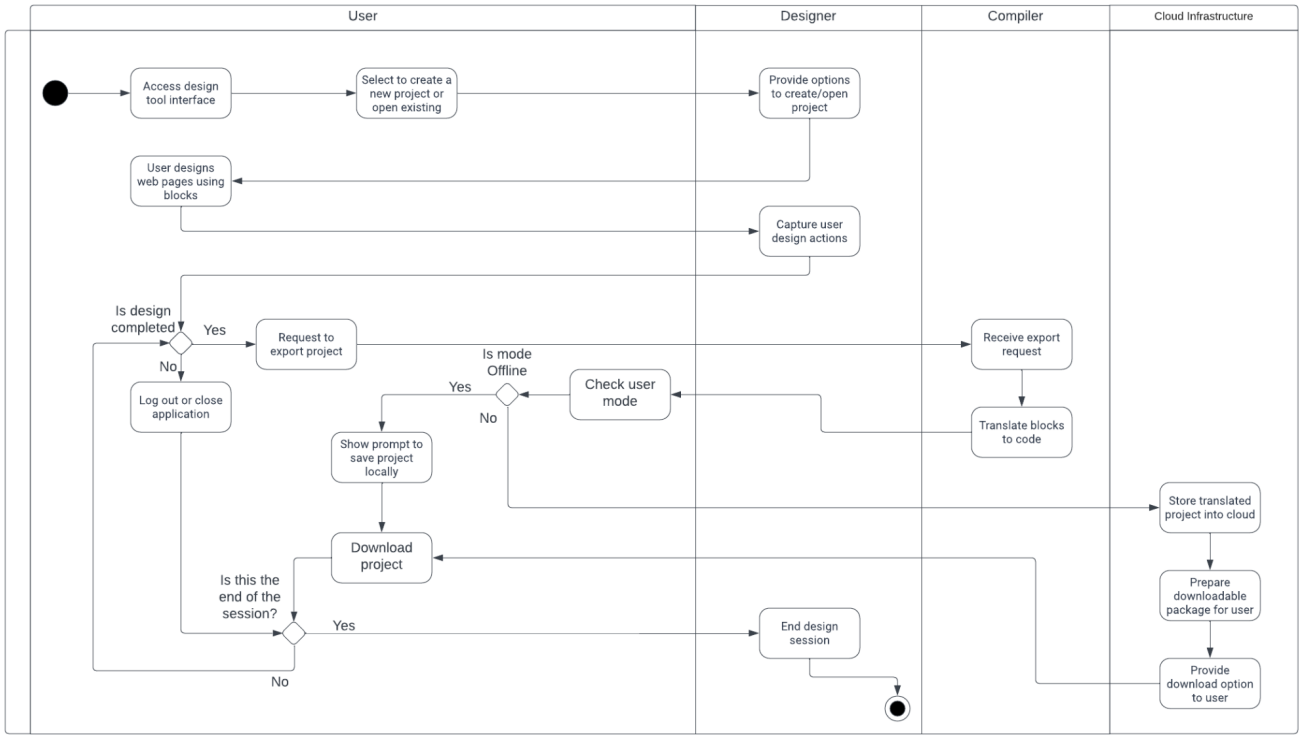

Figure 3: Activity Diagram of the Creating and Exporting a Project (For original resolution, see activity diagram2.png .)

2.5.4 User Interface - Navigational Paths and Screen Mock-ups

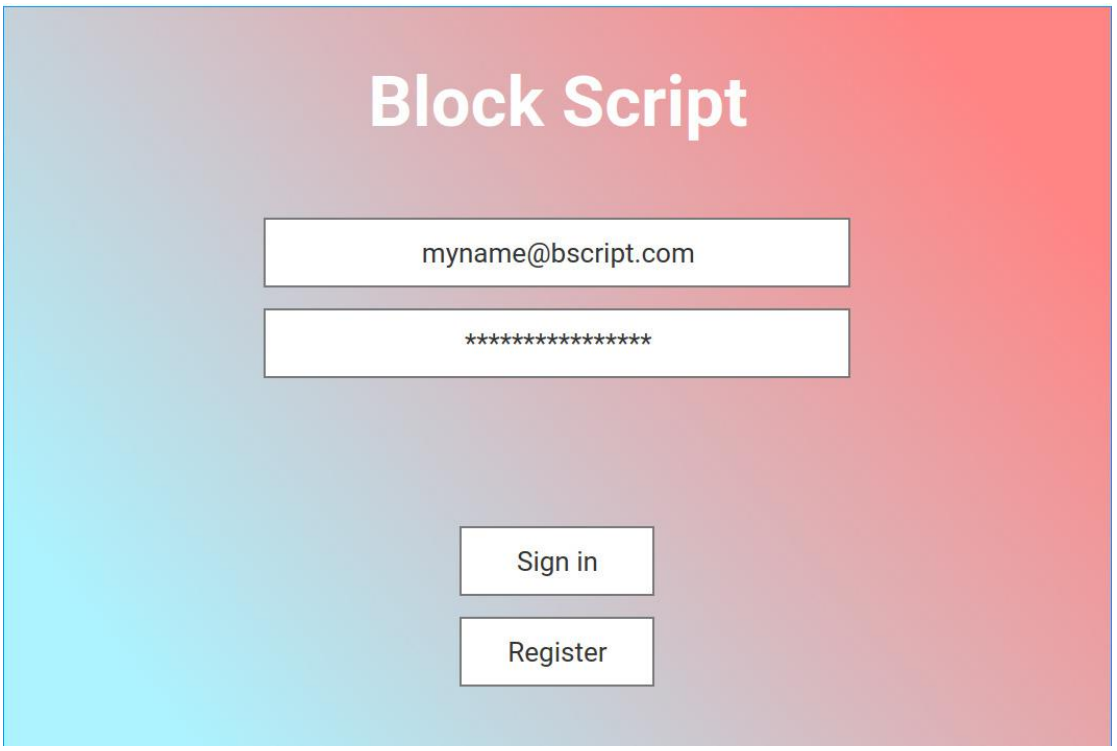

Figure 4: Sign in Screen

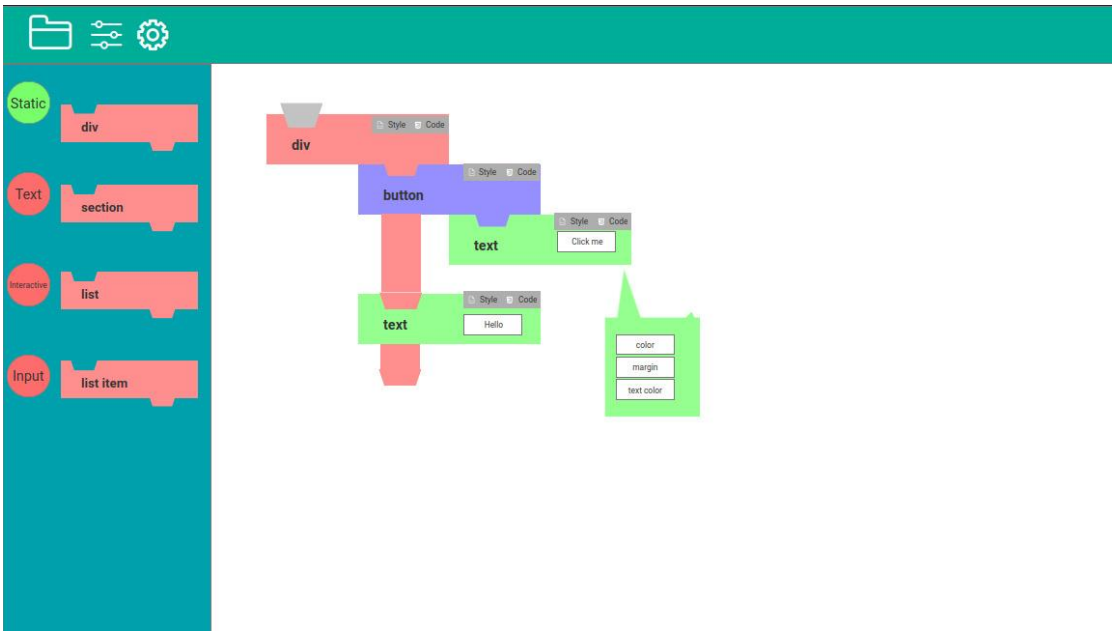

Figure 5: Design Tool Screen while the “ ” block is selected

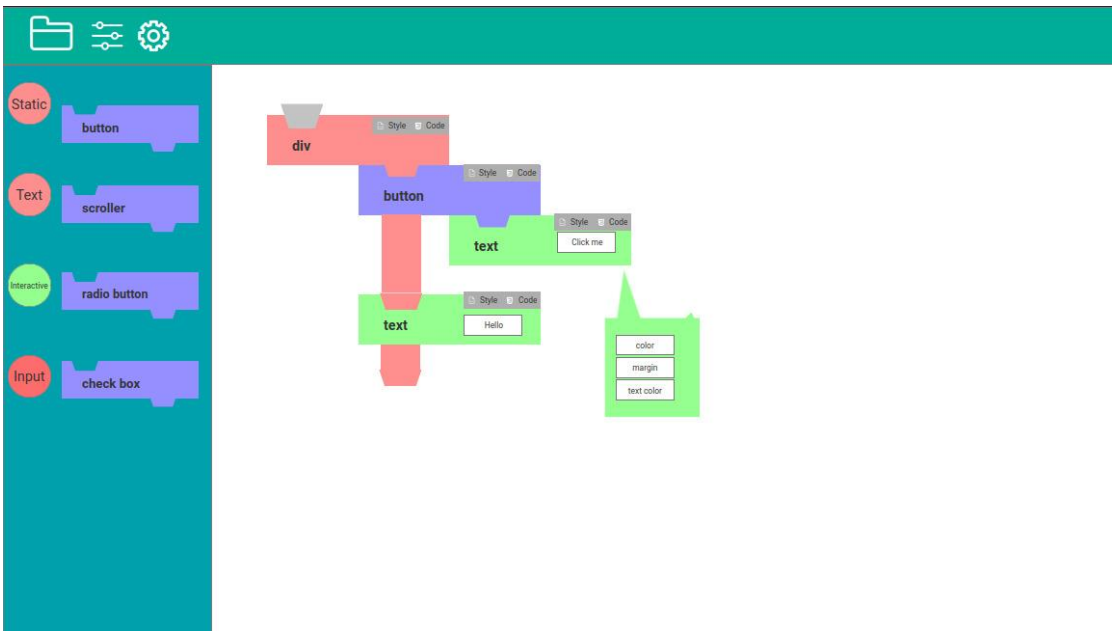

Figure 6: Design Tool Screen while the “<button>” block is selected.

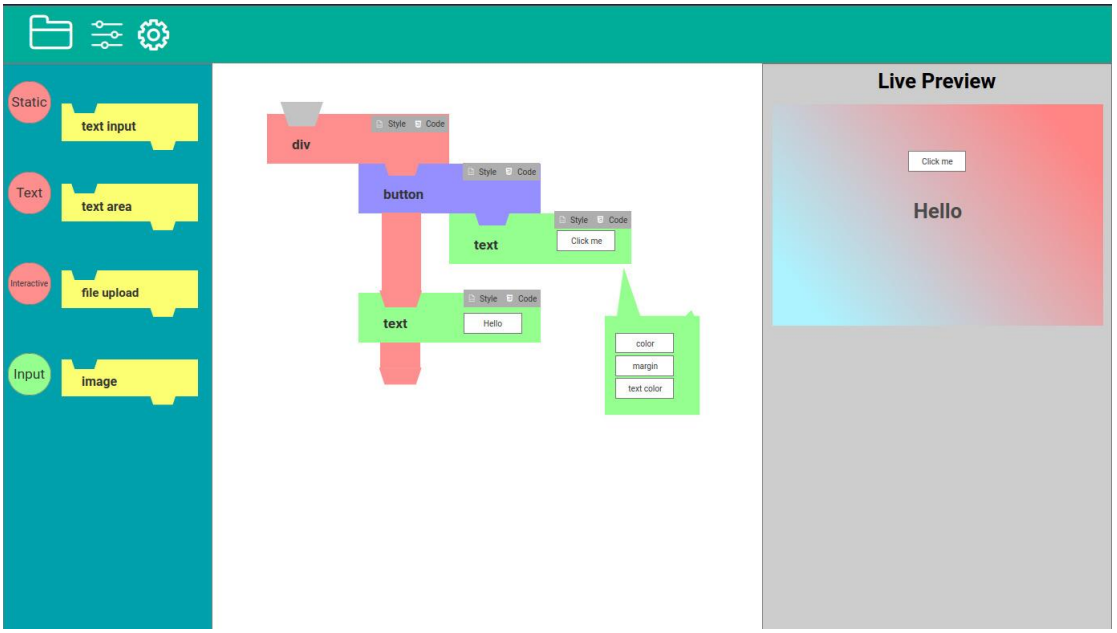

Figure 7: Design Tool Screen while “Preview” mode is on.

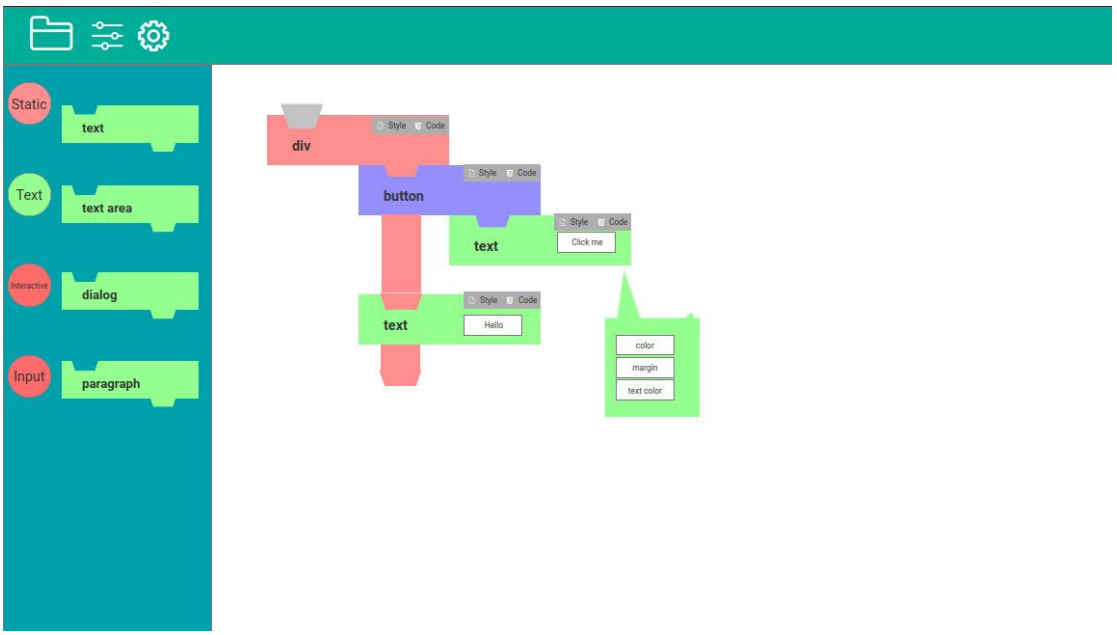

Figure 7: Design Tool Screen while listing different blocks.

3 Other Analysis Elements

3.1 Consideration of Various Factors in Engineering Design

3.1.1 Constraints

3.1.1.1 Economic Constraints

The product will be free for everyone. We believe educational opportunities should be equally accessible by everyone. It is aimed to publish this project as an open-source non-profit product. In addition, publishing it as open-source will open the product to contributions from others, thus, this approach will enhance the product's capabilities.

3.1.1.2 Ethical Constraints

As personal data protection law suggests, any user information and data will not be shared with 3rd party users or companies. Our services will be designed to protect users' data from any leak.

3.1.1.3 Language Constraints

The most common programming languages are in English, and it is expected that there won't be any change on this in the near future. Since this project aims to provide

an education-driven product, we believe learning and practicing the terminology in English is a part of the education towards the children who are new to programming. Therefore, the main language of the product will be English. We will consider supporting different languages in the future, however, the block names will stay in English since it is the common practice.

3.1.2 Educational Purpose

Considering the education purposes, Block Script will provide achievements and statistics for the registered users as an indication of the progress of the user.

3.1.3 Standards

In the documentation, IEEE 830 for the structure and UML 2.5.1 for the diagrams was used.

3.2 Risks and Alternatives

There are multiple alternative tools which provide the service which we want to provide. Some of these alternative tools provide general purpose visual programming languages, such as Scratch, for children but they are designed around web applications. Other alternatives provide visual design for web web development, such as Figma, but they are designed for children. As we are trying to combine these two approaches, and there are not many alternatives doing that, we believe that the risk factor of our project is low.

3.3 Project Plan

Table 1: Factors that can affect analysis and design.

	Effect level	Effect
Public health	0	No effect.
Public safety	3	It has little effect in terms of data.
Public welfare	5	As the welfare level increases people consider more about education.

Global factors	8	BlockScript needs to be designed for global education therefore global factors should be considered.
Cultural factors	6	BlockScript should be understandable by different cultures.
Social factors	9	BlockScript should consider people from different social groups and appeal to them.
Environmental factors	0	No effect.
Economic factors	7	Use of BlockScript in schools related to economic level.

Table 2: Risks

	Likelihood	Effect on the project	B Plan Summary
Time management	Medium	Some extra features that have been planned such as cloud service couldn't be implemented.	Core functions of the system will be prioritized .
Implementation	Low	Failure of implementation on compiler part and connecting it to design tool	Alternative languages will be used. Golang rather than C++ etc.
Distribution of workload	Low	Failure of sharing equally weighted tasks	Tasks will be shared based on knowledge of individuals to ease the process.

Table 3: List of work packages

WP#	Work package title	Leader	Members involved
WP1	Documentation	Mehmet Kaan Örnek	All Members
WP2	Design Tool Front End Development	Göksu Şenerkek	Mehmet Kaan Örnek
WP3	Design Tool Back End Development	Muhammed İkbal Doğan	Khashayar Amini
WP4	Compiler Development	Khashayar Amini	All Members
WP5	Project Demonstration	Khashayar Amini	All Members

WP 1: Documentation			
Start date: October 2, 2023 **End date:** May 5, 2024			
Leader :	Mehmet Kaan Örnek	**Members involved:**	All Members
Objectives: Preparing and maintaining the deliverables documents.			
Tasks: **Task 1.1 :** Preparing Project Specification Document **Task 1.2 :** Preparing Analysis and Requirement Report **Task 1.3 :** Preparing Detailed Design Report **Task 1.4 :** Preparing Final Report			
Deliverables **D1.1:** Project Specification Document **D1.2:** Analysis and Requirement Report **D1.3:** Detailed Design Report **D1.4:** Design Project Final Report			
WP 2: Design Tool Front End Development			
Start date: October 18, 2023 **End date:** February 15, 2024			
Leader :	Göksu Şenerkek	**Members involved:**	Mehmet Kaan Örnek

Objectives: Designing design tool screen mockups, developing and implementation of design tool user interfaces
Tasks: **Task 2.1<Design Tool UI> :** Designing the Design Tool UI and preparing mockups. **Task 2.2<Implementing UI> :** Implementing the designed user interface mockups
Deliverables **D2.1:** Design tool mockups **D2.2:** Design tool user interface implementation

WP 3: Design Tool Back End Development			
Start date: October 18, 2023 **End date:** March 15, 2024			
Leader :	Muhammed İkbal Doğan	**Members involved:**	Khashayar Amini
Objectives: Implementation of design tool backend logic. Development of related APIs and communication between design tool and compiler.			
Tasks: **Task 3.1 <User Registration> :** User registration processes. **Task 3.2 <User Authentication> :** User authentication processes. **Task 3.3 <Development of Functionalities > :** Implementation of the logic of functionalities behind frontend components: Blocks, triggers etc. **Task 3.4 <API development> :** Implementing the communication between compiler and design tool.			
Deliverables **D3.1:** User registration and authentication system. **D3.2:** Design tool related functionalities and components. **D3.3:** Connection between compiler and design tool.			

WP 4: Compiler Development			
Start date: October 18, 2023 **End date:** April 1, 2024			
Leader :	Khashayar Amini	**Members involved:**	All Members
Objectives: Designing and implementing compiler functions.			
Tasks: **Task 4.1 <API development> :** Implementing the communication between compiler and design tool. **Task 4.2 <Compiler functions> :** Implementing compiler functions. **Task 4.3 <Code exportation> :** Logic behind translation of design tool components to html,css and javascript code.			
Deliverables **D4.1:** Exportation from design tool components to code. **D4.2:** Communication between compiler and design tool.			

WP 5: Project Demonstration			
Start date: October 18, 2023 **End date:** December 15, 2023			
Leader :	Khashayar Amini	**Members involved:**	All Members
Objectives: Project presentation to stakeholders			
Tasks: **Task 5.1 <Project presentation> :** Preparing project demonstration. **Task 5.2 <Preparing project mockups> :** Preparing mockups for user interface of design tool.			
Deliverables **D5.1:** Project Presentation. **D5.2:** Project Mockups.			

3.4 Ensuring Proper Teamwork

Each team member within the group will have similar responsibilities. As a requirement of teamwork, everyone will know their responsibilities and fulfill them weekly. A more effective working method will be adopted by dividing the project into

various task distributions. Various applications will be used to monitor the project's progress and stay constantly updated. These applications are:

- Discord: Discord is a chat application used to reinforce teamwork. It will be used to share live developments within the team. Progress on the final project will be easier thanks to some features such as screen sharing and voice chat rooms. In particular, facilitating communication within the team will improve teamwork.
- GitHub: GitHub is a platform that is definitely used to ensure the integrity of the senior project. It offers much convenience in keeping the project constantly updated, updating new codes to be added, and notifying other team members about the added codes.

Peer review can be emphasized to ensure that other team members are constantly monitored and exchanged ideas. With the use of applications like Github and Discord, the flow of the project becomes more efficient and the effects of teamwork are more visible.

3.5 Ethics and Professional Responsibilities

Ensuring the privacy of user information of children is extremely important to us. We will take steps to safeguard data integrity and confidentiality by implementing encryption and security measures. Our application development process will incorporate improvements based on the needs and preferences of all children within our target age group. To create a user tool we will adhere to design principles that promote ease of use for everyone. Additionally we will comply with all requirements, particularly those related to educational software and children's privacy. During the design phase we will also provide resources to help children grasp coding fundamentals easily. By assessing their usefulness, for children we will determine which new features should be included in our application.

3.6 Planning for New Knowledge and Learning Strategies

As today's projects have more innovative requirements, all team members must improve their software skills. While we are fine, especially in applications we constantly use, such as Google Docs, GitHub and Discord, some issues, such as preparing UI/UX design that will appeal to children and advanced Javascript coding, can be challenging. In order to overcome these difficulties, we will access detailed

content prepared on platforms such as YouTube and try to update our knowledge constantly. Especially while learning new content, we will share what we find helpful with each other on our Discord server and try to contribute to teamwork. To prepare our project in coordination with other team members, we will use platforms such as Stack Overflow to solve insoluble code difficulties.

4 References

[1] K. Arslan and M. Akçelik, “Programlama Eğitiminde scratch’in kullanılması: öğretmen Adaylarının Tutum ve algıları,” National Education Academia Journal, https://doi.org/10.32960/uead.455502 (accessed Nov. 15, 2023).

[2] C. Office, “D5 london: Teaching children to code,” GOV.UK, https://www.gov.uk/government/publications/d5-london-summit-themes/d5-london-teaching-children-to-code (accessed Nov. 15, 2023).

[3] “New education policy 2020: Integration of coding and analytical thinking from the schooling level,” Hindustan Times, https://www.hindustantimes.com/education/new-education-policy-2020-integration-of-coding-and-analytical-thinking-from-the-schooling-level/story-G0ihYK8LitcMXmxbwBBd6L.html (accessed Nov. 15, 2023).