EMB00001f4c87c1

|  |
| --- |
|  |
| **한이음 IT멘토링 프로젝트**  [문서 제목 입력] |
|  |

|  |  |
| --- | --- |
| **프로젝트명** | **협업을 위한 웹 기반 통합 개발 환경** |

|  |  |  |  |
| --- | --- | --- | --- |
| **참여인원** | **멘토** | 씨티컴㈜ | 이동기 |
| **멘티** | 한국산업기술대학교 | 최지웅 |
| **멘티** | 한국산업기술대학교 | 김유상 |

**- 내용 목차 -**

[1. 선정배경 4](#_Toc404649904)

[1) 협업이 필요한 작업, SW개발 4](#_Toc404649905)

[2) 이슈가 되고 있는 웹 기반의 클라우드 개발 환경 4](#_Toc404649906)

[3) 고성능의 Back-end 기술 Node.js 4](#_Toc404649907)

[2. 개발 목표 4](#_Toc404649908)

[1) 동시 편집 4](#_Toc404649909)

[2) 채팅 4](#_Toc404649910)

[3) 이용의 편이 4](#_Toc404649911)

[3. 기술 동향 5](#_Toc404649912)

[1) 유사프로젝트 5](#_Toc404649913)

[2) 관련 기술 5](#_Toc404649914)

[(1) 웹 표준 HTML5 5](#_Toc404649915)

[(2) HTML5 캔버스 API 5](#_Toc404649916)

[(3) Node.js 6](#_Toc404649917)

[4. 개발 목표 6](#_Toc404649918)

[1) HTML5를 통한 웹 페이지 제작 6](#_Toc404649919)

[(1) 웹 기반 통합 개발 환경 6](#_Toc404649920)

[(2) 레이아웃 구성 6](#_Toc404649921)

[(3) Code Editor View 6](#_Toc404649922)

[(4) Console View 6](#_Toc404649923)

[2) 안정성 있는 Server/Client간 통신 6](#_Toc404649924)

[(1) 실시간 동기화 6](#_Toc404649925)

[(2) 채팅 6](#_Toc404649926)

[(3) 간단한 로그인 서비스와 코드 저장 7](#_Toc404649927)

[5. 개발 내용 7](#_Toc404649928)

[1) 분석 7](#_Toc404649929)

[(1) 로그인 및 개인 아카이브 7](#_Toc404649930)

[(2) 메인 레이아웃 구성 7](#_Toc404649931)

[(3) 실시간 동기화 8](#_Toc404649932)

[(4) 채팅 8](#_Toc404649933)

[(5) 웹 터미널 9](#_Toc404649934)

[2) 설계 9](#_Toc404649935)

[6. 개발 환경 10](#_Toc404649936)

[1) 개발 OS 10](#_Toc404649937)

[2) 개발 환경 10](#_Toc404649938)

[3) 개발 Tool 10](#_Toc404649939)

[7. 개발 일정 10](#_Toc404649940)

[8. 역할분담 11](#_Toc404649941)

[9. 참고문헌 11](#_Toc404649942)

**- 그림 목차 –**

[그림 1. 유사 프로젝트 6](#_Toc404649949)

[그림 2. HTML5 6](#_Toc404649950)

[그림 3.Node.js 7](#_Toc404649951)

[그림 4. 개인아카이브 8](#_Toc404649952)

[그림 5. main 8](#_Toc404649953)

[그림 6. 실시간 동기화 9](#_Toc404649954)

[그림 7. 텍스트 채팅 9](#_Toc404649955)

[그림 8. 그림판 채팅 9](#_Toc404649956)

[그림 9. 웹 터미널 10](#_Toc404649957)

[그림 10. System Archithecture 10](#_Toc404649958)

[그림 11. Use-case Diagram 11](#_Toc404649959)

[그림 12. 개발 일정 11](#_Toc404649960)

[그림 13. 역할분담 12](#_Toc404649961)

# 1. 선정배경

## 1) 협업이 필요한 작업, SW개발

SW개발은 작은 규모의 프로젝트를 제외하고는 혼자 개발을 진행하기에는 무리가 있다. 즉 팀 단위의 활동이라고 할 수 있다. 큰 규모의 프로젝트에서는 하나의 SW를 함께 제작하는 것이 일반적이다. 여러 사람이 하나의 SW를 보다 원활히 제작하기 위해서 소스코드를 실시간으로 동시에 제작ㆍ수정하며 개발하고자 한다.

## 2) 이슈가 되고 있는 웹 기반의 클라우드 개발 환경

가트너가 매년 발표하는 '10대 전략 기술' 중에 클라우드는 많은 비중을 차지한다. 이처럼 클라우드가 이슈가 되고 있는 세상에서 웹 기반의 클라우드 개발 환경은 개발자에게 있어 꼭 필요한 것이라 생각한다. 집 데스크톱에 개발환경을 모두 구축해놓고, 학교나 회사 등 다른 컴퓨터에서 다시 개발환경을 구축하려면 여간 귀찮은 일이 아니다. 그래서 우리의 프로젝트는 웹 기반의 클라우드 개발환경을 제공하고자 한다. 집, 학교, 회사 어디에서든 인터넷이 되는 곳이기만 하면 집에서 작업하던 그대로 밖에서도 작업을 할 수 있는 클라우드 개발 환경을 제공하여 세상의 변화를 빠르게 수용하고자 한다.

## 3) 고성능의 Back-end 기술 Node.js

Node.js는 구글의 크롬 V8 자바스크립트 엔진을 기반으로 한, 고성능 네트워크 서버이다.

Single Thread 기반의 Event Loop를 기반으로 하고 있으며, File, Network 등에 대해서 비동기 IO 처리를 하는 서버 미들웨어이다. 또한 node.js는 node.js의 모듈중의 하나인 socket.io를 제공한다. 웹 개발이 점점 양방향이 되어가면서, 클라이언트와 connection을 맺고, 여기에 push를 보낼 수 있는 기술이 필요하다. HTML 5의 웹소켓이 좋은 기술이기는 하지만, 브라우져 호환성의 문제가 있는데 반해서, socket.io는 웹소켓을 포함한, AJAX 롤폴링등 여러개의 웹푸쉬를 abstraction하여, 브라우져에 상관 없이 개발자가 쉽게 socket.io API만 이용하면, 푸쉬를 구현할 수 있게 해주며, 싱글 쓰레드 기반의 멀티 플랙싱을 기반으로, 대용량 사용자에 대한 푸쉬 처리를 가능하게 해준다.

# 2. 개발 목표

## 1) 동시 편집

다수의 사용자가 본 프로젝트에서 제공하는 통합 개발환경을 이용하여 동시에 소스코드를 제작할 수 있게 하고, 누가 어느 부분을 수정하고 있는지 실시간으로 알 수 있도록 할 것이다.

## 2) 채팅

프로젝트를 팀 단위로 진행할 때에는 잦은 의사소통이 필요하다. 이를 위해 텍스트 채팅을 제공하고 보다 나은 협업을 위해 그림판 채팅기능을 제공한다.

## 3) 이용의 편이

사용자의 낮은 접근성을 위해 소스코드를 HTML5기반 웹 상에서 개발하는 것은 물론, 웹 터미널 기능을 가능케 하여 사용자가 쉽게 컴파일 및 디버그를 이용가능 하도록 할 것이다. 또한 웹 브라우저만 있으면 언제 어디서나 자유롭게 사용 가능할 수 있도록 할 것이다.

# 3. 기술 동향

## 1) 유사프로젝트

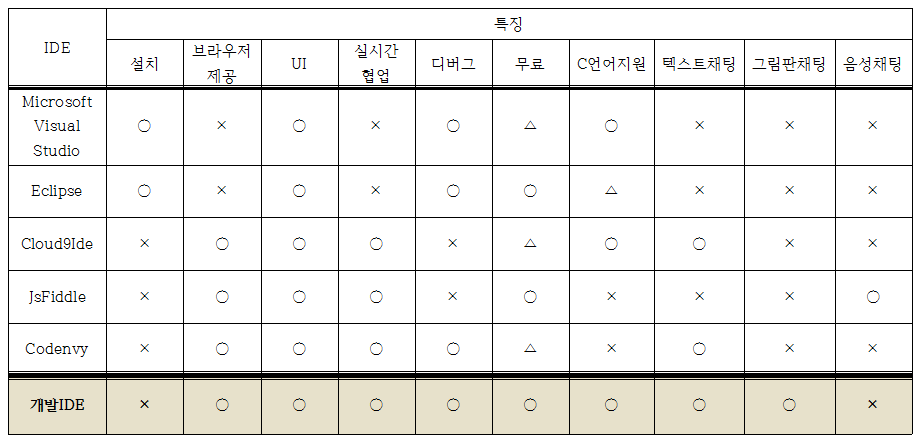


그림 1. 유사 프로젝트

## 2) 관련 기술

### (1) 웹 표준 HTML5

HTML5는 HTML의 차기 주요 제안 버전으로 WWW의 핵심 마크업 언어다. HTML5는 HTML로웹 페이지의 구조를 구성하고 CSS3로 디자인을 하며, Javascript로 동적인 프로그래밍을 하는 전 세계 웹 표준이다.



그림 2. HTML5

### (2) HTML5 캔버스 API

HTML5 캔버스 API를 활용하면 기존의 플래시가 갖고 있던 강점인 화려한 애니메이션과 인터랙티브한 효과를 낼 수 있다. 캔버스 API는 모바일 브라우저를 포함한 최신 브라우저에서 모두 지원하고 있지만, 국내 점유율이 높은 IE8(Internet Explorer 8) 이하에서는 지원하지 않아 아직 국내 웹에서는 활성화되진 않았다. 하지만 최근 캔버스 API를 지원하는 크롬이나 파이어폭스와 같은 브라우저의 점유율이 점점 높아지고 있는 반면, 캔버스 API를 지원하지 않는 구 버전 IE의 점유율은 낮아지고 있는 추세이다. 현재 해외에서는 캔버스 API를 이용한 사이트들이 많이 활성화돼 있으며, 관련된 자바스크립트 라이브러리들도 활발히 개발되고 있다.

### (3) Node.js

node.js는 single thread 기반으로 동작하는 고성능의 비동기 IO (Non-blocking IO)를 지원하는 네트워크 서버이다. 2009년 Ryan Dahl에 의해서 개발이 시작되어 있으며, 현재 수많은 지원 모듈과, 레퍼런스, 에코 시스템을 가지고 있는 오픈 소스 프로젝트 중에 하나이다. Google Chrome V8 엔진으로 개발되어 있으며, 프로그래밍 언어로는 Java script를 사용하며, Event 기반의 프로그래밍 모델을 사용한다. 근래에 들어서 많은 인터넷 기업들이 node.js를 도입하고 있다. Linked in이나 Paypal 그리고 얼마 전에는 그루폰까지 상당 부분의 내부 시스템을 node.js로 전환하였다.



그림 3.Node.js

# 4. 개발 목표

## 1) HTML5를 통한 웹 페이지 제작

### (1) 웹 기반 통합 개발 환경

전 세계 웹표준 Html5에 따라 웹 페이지를 제작하여 어떤 장소에서는 인터넷이 되는 환경이면 개발 가능한 개발환경을 제공하고자 한다.

### (2) 레이아웃 구성

전체적인 레이아웃은 사용자가 친숙한 Visual Studio나 Eclipse의 레이아웃 배치를 참고하여 구성하고자 한다.

### (3) Code Editor View

코드를 작성할 때, 사용자 편의성을 위해 소스코드 강조기능이나 line numbering 기능, Break Point Check 기능 및 다양한 추가 기능을 지원하고자 한다.

### (4) Console View

Editor를 통해서 작성한 프로그램을 컴파일 및 디버깅 하고 그 결과를 출력하는 기능을 구현하고자 한다.

## 2) 안정성 있는 Server/Client간 통신

### (1) 실시간 동기화

사용자가 코드를 치는 내용을 Queue에 저장하고 주기적인 타이머와 Socket.io를 이용하여 동기화를 진행한다. 또한 사용자의 키보드 커서위치를 함께 추적하여 작성중인 위치를 실시간으로 파악하고 같은 부분을 수정하지 않도록 한다.

### (2) 채팅

사용자의 채팅 내용을 Socket.io를 이용하여 Server와 주고받으며 진행한다. 또한 그림판 채팅의 경우 마우스 커서의 위치를 Socket.io를 이용하여 서버와 통신해 동기화를 진행한다.

### (3) 간단한 로그인 서비스와 코드 저장

로그인/로그아웃을 제공하여 적정수준의 보안성을 강화 할 것이다. 또한 서비스의 특성상 사용자가 작성한 소스코드는 모두 서버에 저장된다. 이를 위한 데이터베이스 설계와 구현을 MySQL을 사용하여 경량화된 서버 데이터베이스를 구축할 것이다.

# 5. 개발 내용

## 1) 분석

### (1) 로그인 및 개인 아카이브

간단한 회원가입과 로그인을 통해 본 프로젝트에 접근할 수 있고, 개인 아카이브를 적용하여 프로젝트를 쉽게 만들고 관리할 수 있다.

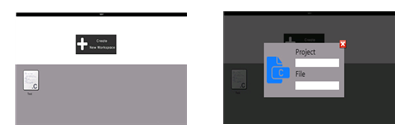


그림 4. 개인아카이브

### (2) 메인 레이아웃 구성

메인 레이아웃은 상단의 간단한 메뉴, 좌측의 파일관리, 우측의 텍스트 및 그림판 채팅, 하단의 웹 터미널, 중간의 편집기로 구성된다.

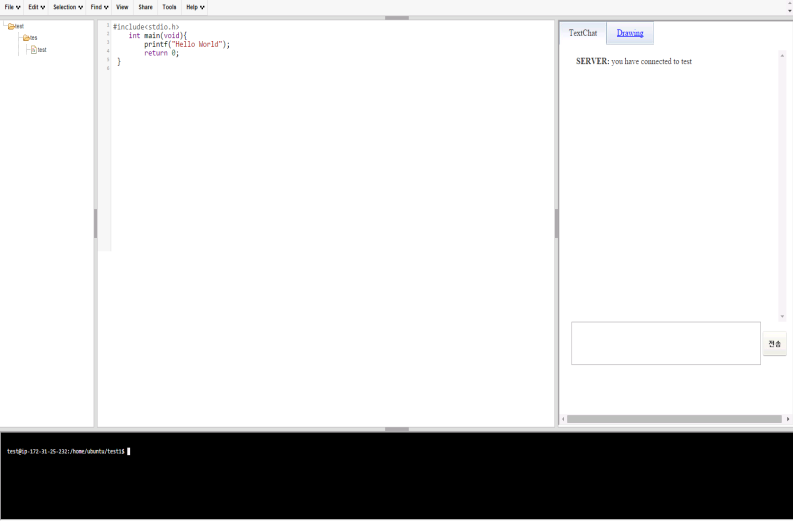


그림 5. main

### (3) 실시간 동기화

업데이트 내용을 반영하는 중에는 다른 업데이트가 일어나지 않도록 하는 Flag를 선언한다. 여러 업데이트 내용들이 충돌하지 않고 순차적으로 반영될 수 있도록 큐를 사용한다. 수시로 업데이트를 수행하기 위해 타이머를 설정한다. Socket.io를 통해 자신이 수정하고자 하는 파일이 맞는지 확인 후 변경된 내용과 커서의 움직임을 큐에 저장하고 동기화 시킨다.

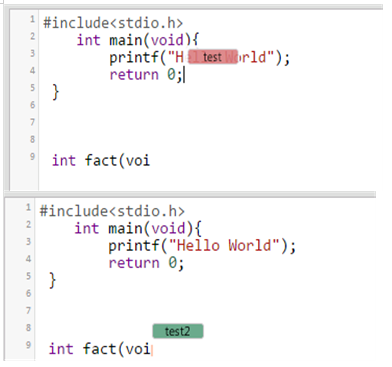


그림 6. 실시간 동기화

### (4) 채팅

초기 Project 생성 시에 사용자 ID와 Project명으로 구성된 채팅방을 생성하고 Socket.io를 이용하여 메시지가 입력될 때, 다른 사용자가 접속할 때 메세지를 주고받는다. 그림판 채팅의 경우에는 사용자가 클릭하는 모든 좌표를 주고받아 동기화 시킨다.

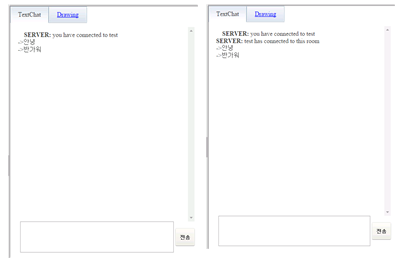


그림 7. 텍스트 채팅

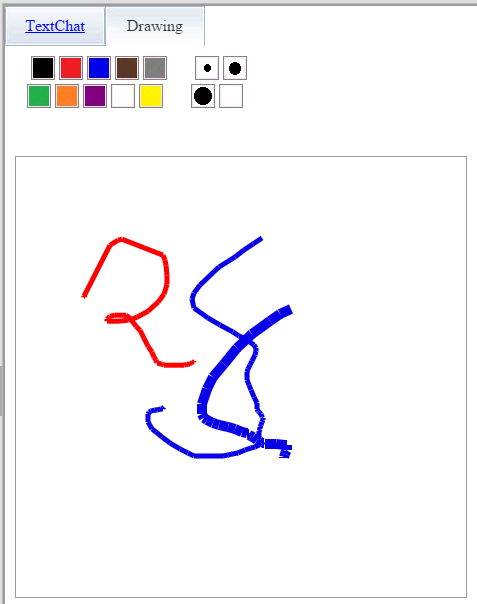


그림 8. 그림판 채팅

### (5) 웹 터미널

로그인을 하면 회원가입 시 생성된 계정으로 웹 터미널이 접속된다. 서버의 한정된 기능을 제공하며 GCC와 GDB를 이용하여 컴파일 및 디버깅을 진행할 수 있다.

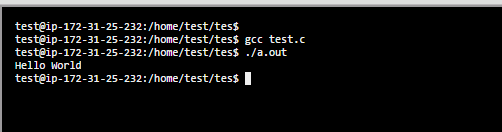


그림 9. 웹 터미널

## 2) 설계

EMB00001f4c8b26

그림 10. System Archithecture

전체적인 시스템 구조는 위와 같다. 크게 다음과 같이 나눌 수 있다. Node.js Server, MySQL(DB), Client. 사용자는 회원가입을 통해 웹 계정과 서버 계정의 두 가지 계정을 갖게 된다. 로그인 후 IDE의 아카이브를 이용하여 프로젝트를 생성한다. 생성된 프로젝트 및 파일은 서버의 사용자 계정안 폴더 및 파일로 생성된다. 이 파일에 접근하여 소스코드를 제작 및 수정할 수 있다. 또한 Socket.io를 이용하여 같은 페이지에 접속 한 사용자들과 작업 중인 소스코드를 보면서 실시간으로 협업하며 수정할 수 있다. 실시간으로 수정하는 부분은 키보드 커서 위치를 추적하여 각 사용자의 ID와 함께 모두에게 보여지게 된다. 또 웹 채팅기능을 이용하여 같은 소스코드를 수정 중인 사용자간에 의사소통을 제공하고, 그림판 채팅을 통해 보다 원활한 의사소통을 진행할 수 있다. 작성을 완료한 소스코드는 웹 터미널을 이용해 컴파일 및 디버깅을 진행할 수 있다.

EMB000014201601

그림 11. Use-case Diagram

# 6. 개발 환경

## 1) 개발 OS

Windows 7, Ubuntu 12.04 LTS

## 2) 개발 환경

HTML5, Javascript, CSS3, Node.js, MySQL, Amazon EC2

## 3) 개발 Tool

Putty, Visual Studio, EditPlus

# 7. 개발 일정

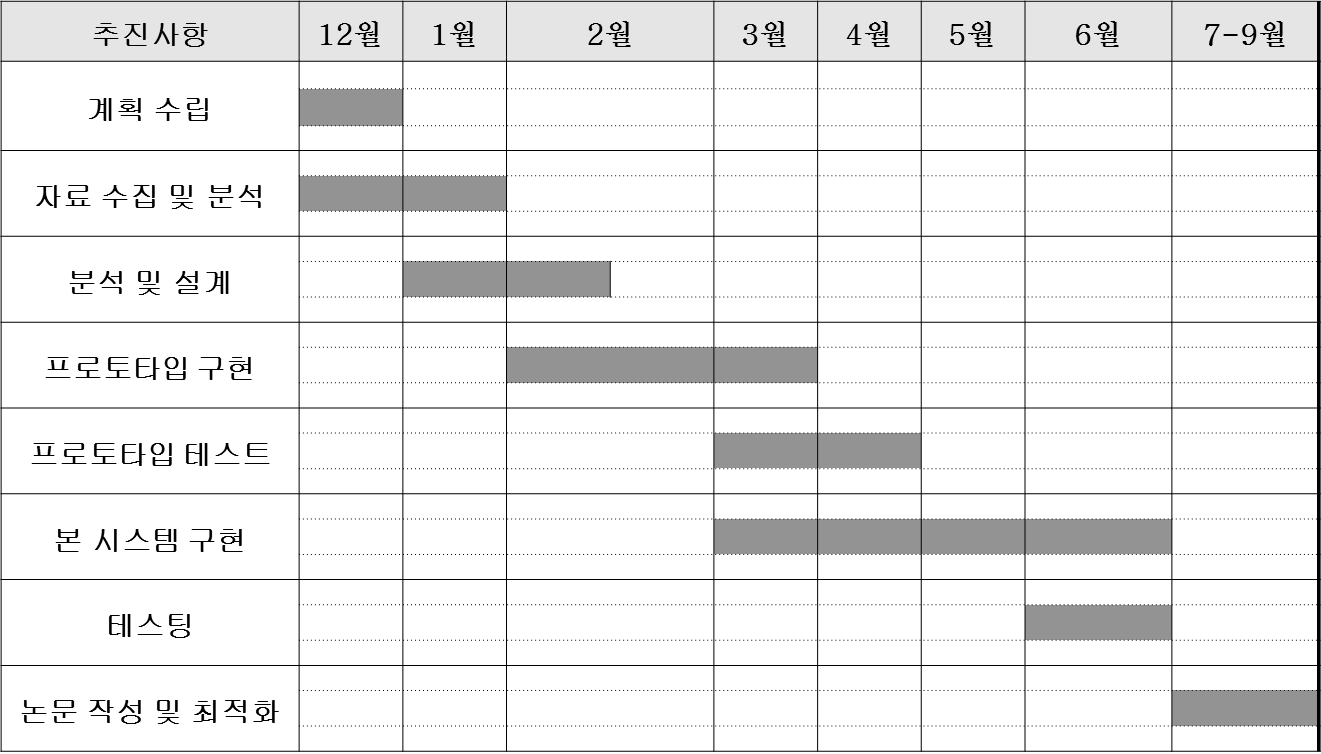


그림 12. 개발 일정

# 8. 역할분담

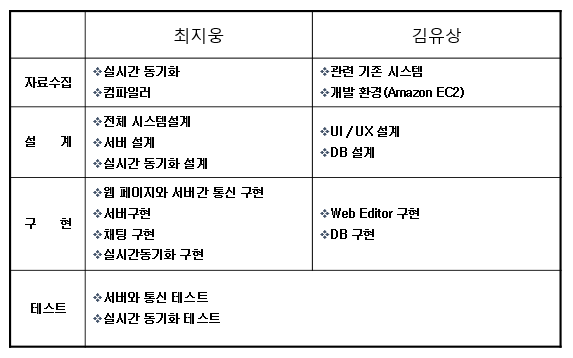


그림 13. 역할분담

# 9. 참고문헌

[1] GitHub,

“http://ko.wikipedia.org/wiki/%EA%B9%83%ED%97%88%EB%B8%8C”

[2] Node.js 위키백과,

“http://ko.wikipedia.org/wiki/Node.js“

[3] Node.js 네이버 카페,

“http://cafe.naver.com/buldon/6252”