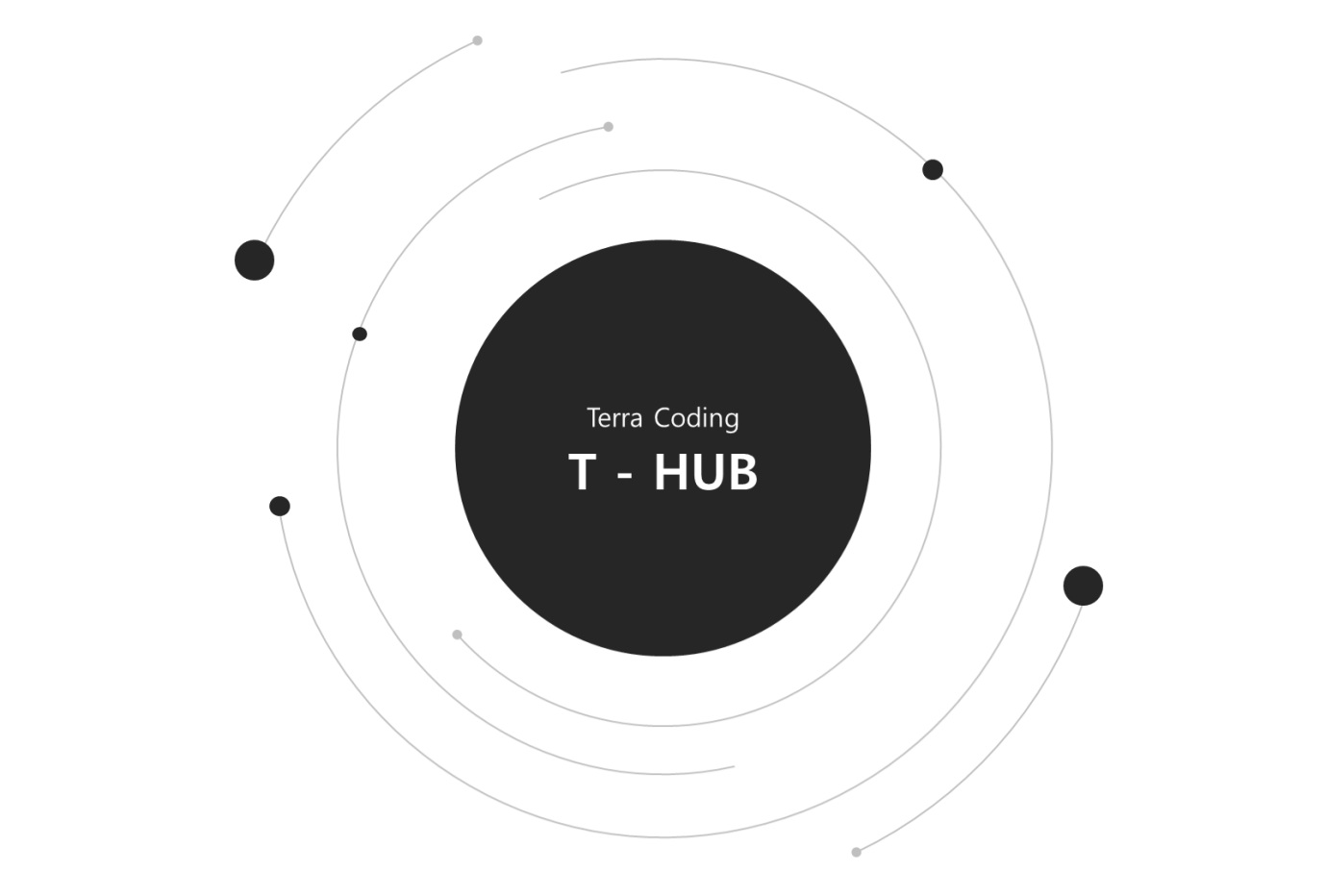
Capstone Design Project

코딩 초보자를 위한 웹 플랫폼

**“T-HUB”**



**Team “Terra Coding”**

기획 제안서

|  |  |
| --- | --- |
| 201222702 | 하재준 |
| 201423044 | 이승환 |
| 201423046 | 박 진 |
| 201423053 | 이경재 |
| 201423051 | 엄준영 |

**Index**

|  |  |
| --- | --- |
| **1 프로젝트………………………………………………………………………………..** | **7** |
| 1.1 동향……………………………………………………………………………. | 7 |
| 1.2 문제 정의……………………………………………………………………... | 8 |
| 1.2.1 부족한 코딩 사교육 대체 방안…..…………………………………. | 8 |
| 1.2.2 코딩 외 진입 장벽……..………….…………………………………. | 8 |
| 1.2.3 알고리즘 풀이만이 피드백의 정도(正道)인가?………….............. | 8 |
| 1.3 타겟……………………………………………………………………………. | 9 |
| 1.3.1 코딩의무교육 대상자……….………………………………………. | 9 |
| 1.3.2 프로그래밍 입문자………….………………………………………. | 9 |
| 1.4 가치……………………………………………………………………………. | 9 |
| 1.4.1 시간 단축……….……………………………………………………. | 9 |
| 1.4.2 자유로운 개발 환경…………………………………………………. | 10 |
| 1.5 핵심 기술 요소……………………………………………………………….. | 10 |
| 1.5.1 컴파일 결과에 따른 피드백……...…………………………………. | 10 |
| 1.5.2 프로젝트 단위의 코드 작성……...…………………………………. | 10 |
| 1.5.3 사용자 정의의 가상 데이터베이스...………………………………. | 11 |
| **2** **기존 사례 분석………………………………………………………………………...** | **11** |
| 2.1구름IDE……………………………………………………………………….. | 11 |
| 2.2 Baekjoon Online Judge….………………………………………………….. | 11 |
| 2.3 Insight 및 차별성………….………………………………………………….. | 11 |
| **3 개발 내용……………………………………………………………………………….** | **12** |
| 3.1 목표……………………………………………………………………………. | 12 |
| 3.2 주요FEATURES………………………………………...…………………...... | 12 |
| 3.3 USE CASES………………………………………………………….……….... | 14 |
| 3.3.1 Use Case 1......................................………………………………. | 14 |
| 3.3.2 Use Case 2......................................………………………………. | 16 |
| 3.3.3 Use Case 3......................................………………………………. | 16 |
| 3.3.4 Use Case 4......................................………………………………. | 18 |
| **4 설계(DESIGN).....................................................................................................** | **20** |
| 4.1 OVERALL 시스템 구조…………………………………………………..……. | 20 |
| 4.1.1 User Interface……………………………….………………………. | 20 |
| 4.1.2 Server………………...………………………………………………. | 20 |
| 4.2 SW ARCHITECTURE……………………………………………….………….. | 21 |
| 4.2.1 Client Side…………...………………………………………………. | 21 |
| 4.2.2 Server Side……..……………………………………………………. | 21 |
| 4.3 LIST OF COMPONENTS……...………………………………………………... | 22 |
| **5 개발 관리……………………………………………………………………………….** | **23** |
| 5.1. 개발 환경 / 언어 / FRAMEWORK 활용 / 운영방안………………………….. | 23 |
| 5.2 리스크 분석 및 회피 방안………………………………………………….... | 23 |
| 5.3 개발 일정……………………………………………………………………... | 25 |
| 5.4 비용 분석……………………………………………………………………… | 26 |
| 5.5 소스 코드 관리 및 CI 툴 운영 방안…………………………………………. | 27 |
| 5.6 CODE CONVENTION…………………………………………………..………. | 27 |
| **6 업무 분장 계획………………………………………………………………………...** | 28 |
| 6.1 FEATURE / 컴포넌트 기반……………………………………………………. | 28 |
| 6.2 기타과제관리/DEPLOY/발표 등 부가적 업무 분장……………………….... | 29 |
| **7 DEPLOY 계획(DEMO)........................................................................................** | **29** |
| 7.1 데모 시나리오………………………………………………………………... | 29 |
| 7.1.1 데모 시나리오 1……....……..………………………………………. | 30 |
| 7.1.2 데모 시나리오 2……....……..………………………………………. | 30 |
| 7.1.3 데모 시나리오 3……....……..………………………………………. | 31 |
| 7.1.4 데모 시나리오 4……....……..………………………………………. | 32 |
| 7.2 KPI(KEY PERFORMANCE INDICATOR)........................................................ | 33 |
| 7.3 FUTURE PLAN…………………………………………………………………. | 33 |

Figure Index

|  |  |
| --- | --- |
| Figure 1. 코딩 사교육 문제 2017년 관련기사……………………………………………… | 7 |
| Figure 2. T-HUB Use Case Diagram ……………………………………………………..… | 14 |
| Figure 3. System Architecture ………………………………………………………………. | 20 |
| Figure 4. Software Architecture …………………………………………………………….. | 21 |
| Figure 5. Demo scenario 1 flowchart ……………………………………………………….. | 30 |
| Figure 6. Demo scenario 2 flowchart ……………………………………………………….. | 31 |
| Figure 7. Demo scenario 3 flowchart ……………………………………………………….. | 32 |
| Figure 8. Demo scenario 4 flowchart ……………………………………………………….. | 32 |

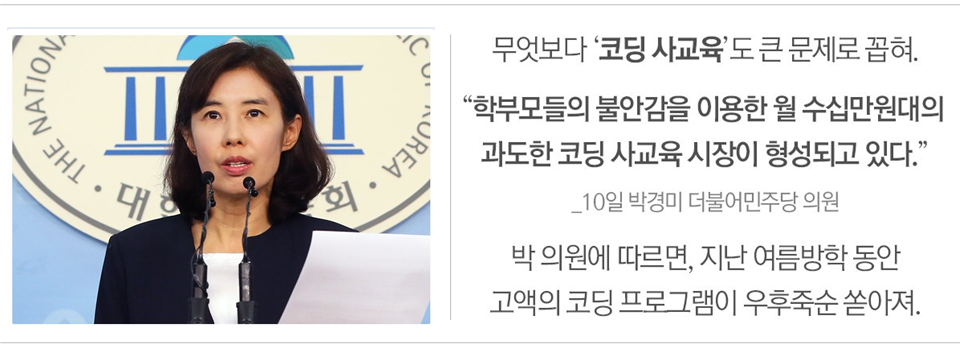
Table Index

|  |  |
| --- | --- |
| Table 1. Main feature…………………………….…………………………………………… | 14 |
| Table 2. Use Case 1……..……………………….…………………………………………… | 15 |
| Table 3. Use Case 2..…………………………….…………………………………………… | 16 |
| Table 4. Use Case 3..…………………………….…………………………………………… | 17 |
| Table 5. Use Case 4..…………………………….…………………………………………… | 18 |
| Table 6. System Layer…..……………………….…………………………………………… | 21 |
| Table 7. List of components………….………….…………………………………………… | 22 |
| Table 8. Development environment…………….…………………………………………… | 23 |
| Table 9. Risk Analysis and Avoidance Measures …………………………….…………… | 23 |
| Table 10. Development Schedule ………………………….……………...……………..… | 25 |
| Table 11. Expense Analysis ……………………….………………………………………… | 27 |
| Table 12. Source Code Management ………………………….………...………………… | 27 |
| Table 13. Continuous Integration ………………………….………………………………... | 27 |
| Table 14. Code Convention of PYTHON ………………………….……………………….. | 28 |
| Table 15. Division of works ………………………….………………………………………. | 28 |
| Table 16. Additional division of works ………………………….…………………………... | 29 |
| Table 17. Demo scenario 1………………………….……………………………………….. | 30 |
| Table 18. Demo scenario 2………………………….……………………………………….. | 31 |
| Table 19. Demo scenario 3………………………….……………………………………….. | 32 |
| Table 20. Demo scenario 4………………………….……………………………………….. | 33 |
| Table 21. KPI………………………….……………………………………………………….. | 33 |
| Table 22. Future plan………………………….……………………………………………… | 33 |

1 프로젝트 개요

1.1 동향

최근 들어 엄청난 영향력을 가진 유명인들이 ‘4차 산업 혁명’이라는 단어를 사용하기 시작하면서 교육 업계에도 적지 않은 영향을 주게 되었다. 세계경제 포럼에서 클라우스슈밥이 이 단어를 사용 하고난 뒤[[1]](#footnote-1) 엄청난 파장을 불러일으켰고, 오바마 대통령 또한 공적인 자리에서 4차 산업 혁명을 언급, 코딩 교육을 장려하면서, 평소에 자주 사용하지 않던 ‘코딩 교육’이라는 단어가 이제는 일상생활에서 아무렇지 않게 사용되기 시작했다. 상황이 이와 같이 흘러다가 보니, 국내에서도 코딩교육이 의무화 되었을 뿐만 아니라, 관련 사교육 시장 또한 심상치않게 흘러가고 있다.

[Figure 1. 코딩 사교육 문제 2017년 관련 기사 ][[2]](#footnote-2)

위 기사는 2017년 10월 13일 News1에서 발행한 기사를 발췌한 것이다. 이 기사에 따르면 현재 한국에서 과도한 코딩 사교육 시장이 형성되고 있다고 한다. 이러한 사회 문제는 코딩 교육 의무화와 더불어 한국의 극성맞은 학부모들도 한 몫을 했을 것이다. 하지만 지금 당장 이러한 사회문제를 해결하기 위한 비책이 있어 보이지 않는다. 정부가 제시한 코딩 사교육 금지법[[3]](#footnote-3) 이 이러한 사회문제를 해결하기 위해 도움이 되겠지만, 결국 근본적인 해결책이 되지는 못할 것이 분명하다.

1.2 문제 정의

1.2.1 사교육

국가에서 의무로 지정한 코딩교육의 실행 체계가 미비한 상황일 뿐만 아니라, SW교육 체계에 맞는 평가 틀도 마련되지 않아 암기 위주 학습이 이뤄질 것이란 우려도 적지 않은 상황에서 과연 얼마나 학생들의 코딩 역량을 끌어 올릴 수 있을지 의문이다. 그렇다고 이러한 문제점들을 적게는 수십에서 많게는 수백만원을 호가하고, 학생들의 학업부담을 가중시키는 사교육으로 해결하자니 문제가 많다. 소프트웨어 교육 의무화 시행을 앞두고 코딩 학원, 코딩 과외 등의 사교육 과열로 번지기보다는 코딩을 배우려는 사람들을 위한 환경이 마련돼야할 것이다.

1.2.2 코딩 외 진입 장벽

코딩이라는 새로운 학문을 익히는것이 쉽게 느껴지는 사람은 선택받은 극소수의 사람들을 제외하고는 거의 없을 것이다. 쉽지 않은 코딩을 준비 과정 때문에 흥미를 잃거나 방향성을 잃는 다면 참으로 안타까운 일이 아닐 수 없다. 코딩 초심자 때를 생각해 보면 컴파일 버튼이 어디 있는지 조차 몰라서 많은 시간을 환경 설정을 익히는데 보냈고, JAVA를 설치하고 환경변수를 설정한다는 것이 무엇인지, x86과 x64의 JAVA version에 따라서 환경 변수 경로가 어떻게 달라지는지 조차 몰라서 한동안 JAVA를 배우는 것을 뒤로 미루기도 하였다. 이러한 환경 설정을 편리하게 만들어 주어서 코딩 입문자가 오직 코딩에만 집중할 수 있는 환경을 제공하는 것이 바람직할 것이다.

1.2.3 알고리즘 풀이만이 피드백의 정도(正道)인가?

유사한 몇몇 웹 IDE 서비스를 관찰해본 결과, 알고리즘 문제를 출제하는 출제자와, 알고리즘을 구현하여 테스트 셋을 만족하는 결과를 얻을 경우 정답으로 처리하는 기능이 제공되고 있다. 물론 알고리즘이 코딩 실력향상에 도움을 주는 것이 맞지만, 실제 현업에서 코딩을 할 때 이미 회사에서 기존에 사용하던 알고리즘을 불러다 쓰는 경우가 대부분일 것이고, 알고리즘을 초기단계부터 설계하는 직군이 많지는 않을 것이다. 때로는 회사에서 제공하는 가이드 라인에 맞춰서, 공동 작업에 최적화된 코딩하는 방법을 익히는 것 또한 프로그래머로서의 훌륭한 자질 중 하나로 손 꼽히기도 한다. 그러므로 이러한 코드 협업 능력을 평가해줄 수 있는 피드백 기능이 필요해 보인다.

1.3 타겟

1.3.1 코딩의무교육 대상자

2018학년도부터 초등학교, 중학교의 코딩의무교육이 시작되고 고등학교의 경우에는 선택할 수 있겠금하여 본격적인 코딩교육이 시작된다. 초, 중, 고등학교의 코딩교육은 전세계적인 추세인데, 처음 코딩에 입문하려는 학생들에게 가장 중요한것은 거부감없는 환경이 가장 큰 변수라고 생각된다. 처음부터 책에 있는 예제들을 배우고 연습하기도 전에 실습해보기 위한 환경설정을 하는데에서 많은 시간을 투자하고 또 심지어 환경설정에 실패하면 시작도 하기전에 코딩교육에 흥미를 잃을 수 있는 요소가 될 수 있다. 우리 T - HUB는 코딩에 처음 입문하려는 사람들에게 환경설정처럼 그 어떤 진입장벽도 생기지 않게하여 시작부터 코딩에 흥미를 잃을수도 있는 요소를 제거하며, 프로젝트 관리를 통해 코딩을 하면서 협업의 재미도 느낄 수 있게할 수 있는 점도 제공한다.

1.3.2 프로그래밍 입문자

어느정도 소프트웨어툴을 사용해본 대학 전공자들도 어느한 과제를 수행하기 위해 필요한 소프트웨어를 설치 및 원하는 환경설정을 하기 위해서 상당한 시간과 노력을 소모한다. 이와같은 관점에서 프로그래밍 입문자의 경우 대부분 처음 접하는 언어가 C언어, 자바등이 될텐데 이 프로그래밍 입문자들은 이 C언어와, 자바를 연습하기 위해 Visual Studio나 이클립스를 설치하고 환경설정을 해야만한다. 특히 자바 이클립스의 경우 여러 환경변수도 설정하여 프로그래밍을 연습하려는 초심자들의 경우 상당한 시간과 노력의 투자가 필요한 경우가 빈번한데, 우리 T-HUB는 간단히 웹페이지 접속을 통해 갖춰진 환경에서 실습만 하면 되므로 이제 막 프로그래밍에 입문하려는 사람들에게는 최적의 환경을 제공한다.

1.4 가치

1.4.1 시간 단축

현재 프로젝트에서 내세우고 있는 컨셉은 편리함이다. 처음 프로그래밍을 시작할때 가장 시간이 많이 드는 부분은 개발 환경 설치이다. 각 개발환경 사이트에 직접 들어가 설치 파일을 다운 받고, 설치 방법을 따로 찾아보고, 또 설치 과정에서 발생하는 수많은 오류들로 인해 많은 시간을 소비하게 된다. 설치를 했더라도 자신에게 알맞는 코딩 작업 환경을 만들기 위해서는 또 다른 문제가 생기게 된다. 이처럼 개발 환경 내의 수많은 기능들에 익숙해질려면 추가로 시간을 소비하게 된다. 하지만 T-HUB가 제공하는 웹 컴파일 기능의 경우, 웹사이트에 접속만 하여, 바로 코딩을 할 수 있는 장점이 있다. 이를 통해 처음 프로그래밍을 시작하는 사람들도 시간을 아낄 수 있고, 흥미를 잃지 않고 코딩에 집중 할 수있는 환경을 제공해줄 것이다.

1.4.2 자유로운 개발 환경

다양한 컴퓨터언어에 대해서 별도의 환경설정 필요없이 인터넷이 되는 환경에서 언제든 접속해 편리하게 코딩을 할 수 있도록 사용자가 제공 받을 수 있다면 앞으로 코딩을 처음접하게 될 코딩의무교육대상자, 프로그래밍 입문자는 물론 간단하게 다른언어를 사용해보고 싶은 기존 프로그래머들에게도 좋은 플랫폼이 될 것이다. 저희 T-HUB는 언제, 어디서든 만족스런 서비스를 제공할 수 있는 웹 플랫폼을 제작하여 궁극적으로는 저희 웹 플랫폼을 사용하려는 사용자들에게 최상의 서비스를 제공하는 자유로운 환경을 제공하는것이 궁극적인 목표이다.

1.5 핵심 기술 요소

1.5.1 컴파일 결과에 따른 피드백

우리 T-HUB는 정적 분석 도구인 SonarQube를 활용하여 코드 자체 퀄리티를 평가해줘 보다 깔끔하고 완벽한 코드를 완성할 수 있도록 도와준다. 평가기준은 사용자가 완성한 코드를 대상으로 하는데, 예를들어 import하였는데 쓰이지 않거나, 썼더라도 중복된 경우 이와 관련된 오류메시지를 알려 코드를 수정하도록 권해준다.

1.5.2 프로젝트 단위의 코드 작성

코드를 작성할 때 코드를 한 파일에 작성 하는 것은 후에 유지 보수의 불편함을 초래한다. 이의 단점을 극복하기 위해 T-HUB는 코드를 프로젝트 단위로 작성 및 컴파일 할 수 있다.

1.5.3 사용자 정의의 가상 데이터베이스

보통 코딩을 하면 데이터베이스를 사용하는 것은 불가피 하다. 우리 T-HUB에서는 사용자들이 데이터베이스를 쉽게 사용할 수 있게 가상 데이터베이스 기능을 제공할 것이며, 가상 데이터베이스를 쉽게 관리 할 수 있도록 시각적인 관리 페이지를 개발할 것이다.

2 기존 사례 분석

2.1 구름IDE

구름IDE은 오픈소스 소프트웨어 형태로 2008년에 개발 되었고, 2017년 4월달에 클라우드 통합 개발 환경을 정식 출시하였다. 서비스로는 웹 개발환경을 다양한 언어에 맞춰 제공하고 있고, 웹 터미널인 리눅스 시스템을 제공해줌으로써 자신의 컴퓨터처럼 필요한 환경을 원하는대로 설치할 수 있게 도와주고 있다. 또한, 프로그래밍을 할 때 있어서 중요한 다른 개발자들과의 협업 시스템을 동시 편집과 채팅을 통해 제공해주고 있고, GIT과 SVN 지원도구와 함께 다양한 환경을 제공해주고 있다. 하지만 피드백을 따로 제공해주지 않고 있고, 처음 프로그래밍을 접하는 사람들에게는 리눅스 터미널과 같은 어려운 인터페이스를 가지고 있다.

2.2 Baekjoon Online Judge

Baekjoon Online Judge는 2010년 3월 19일부터 운영을 시작한 Online Judge 이다. 프로그래밍 문제를 해결한 후에 이를 온라인으로 채점해 볼 수 있는 온라인 시스템으로서 코드를 컴파일하고 실행하며, 미리 작성된 데이터로 테스트할 수 있게 해준다. ICPC 대회를 준비하는 사람 외에도 프로그래밍 공부를 하는 모든 사람들을 위해서 60가지의 프로그래밍 언어를 지원하고, 2017년 기준으로 약 13,000개의 문제를 제공한다. 하지만 사용자는 문제 및 알고리즘을 풀기 위한 코딩만을 할 수 있고, 자신만의 프로젝트를 만들어 프로그래밍을 한다거나 프로젝트 단위로 코드를 관리를 할 수가 없다.

2.3 Insight 및 차별성

T-HUB에서는 타 웹 플랫폼과 동일하게 기본적인 웹 IDE 기능을 가지지만, 다른 플랫폼에는 없는 몇 가지 차별점들이 있다.

먼저, 구름 IDE의 경우에는 데이터베이스 관리를 콘솔로 하게 되는데 이는 소프트웨어를 접해보지 않은 초보자들에게 거북함을 준다. 그래서 T-HUB에서는 초보자들이 데이터베이스를 쉽게 접할 수 있게 시각적인 데이터베이스 관리 페이지를 제공할 예정이다.

또한 우리는 컴파일 후 단순히 출력에서 끝내는 것이 아니라 컴파일 후, 코드 자체에 대한 피드백을 제공함으로써 교육을 받는 사람이 더 효율적인 코드를 작성하게 돕는다.

3 개발 내용

3.1 목표

본 소프트웨어는 다음과 같은 개발 목표를 가진다.

[C, JAVA, PYTHON 을 지원하는 Web IDE 개발]

많이 쓰이는 언어 3가지: C, JAVA, PYTHON을 지원하는 Web IDE를 개발한다. 개발 하려는 Web IDE의 기본 기능으로는 컴파일, 결과 확인 등이 있다.

[JAVA에 대해 컴파일 결과를 피드백 해주는 피드백 기능 개발]

코드를 컴파일 후, SonarQube의 기능을 활용해 컴파일 한 코드의 품질을 체크함으로써 피드백을 제공한다. 현재 JAVA언어만 피드백 기능을 제공할 예정이다.

[간단한 문제 생성 및 채점 시스템 개발]

교육을 위한 문제 생성 시스템을 개발한다. 문제 생성은 관리자로 로그인을 해야 생성이 가능하며, 여기서 문제의 제시문과 채점을 위한 테스트 케이스들을 입력하게 한다. 또한 기본적으로 간단한 문제를 5~10개 정도 추가 할 예정이다.

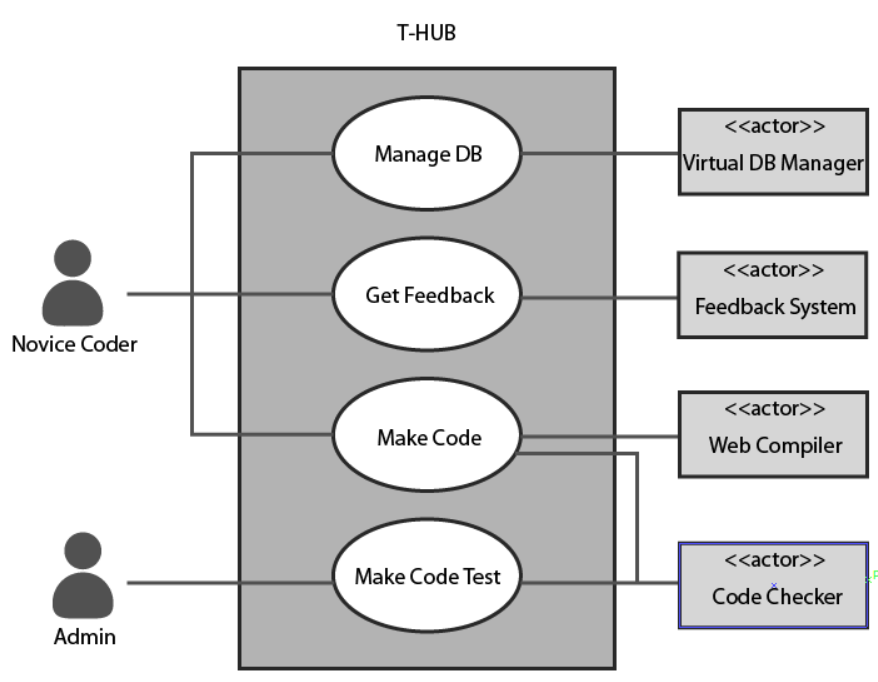
[사용자가 데이터베이스를 쉽게 관리할 수 있도록 시각적인 데이터베이스 관리페이지 개발]

구름 IDE에서의 데이터베이스 기능은 콘솔로 접근 해야하는 단점이 있다. 이는 초보자들이 접근하기에는 거부감을 느낄 것이며 이를 위해 데이터베이스 관리를 위한 시각적인 웹 페이지를 구현할 예정이다.

3.2 주요 FEATURES

|  |  |
| --- | --- |
| **Features** | **Description** |
| 학사 정보로 로그인 | 사용자는 별도의 회원가입 없이도 자신이 이용하는 기존의 학교 계정을 이용하여 T-HUB 시스템에 로그인할 수 있다.  우리 시스템은 아주대학교 로그인 시스템과 연동되었기 때문에 id와 password 입력 폼에 아주대학교 계정 정보를 입력할 경우 로그인 된다. |
| 프로젝트 관리하기 | 사용자는 프로젝트 단위로 코드를 관리할 수 있다. 각 프로젝트는 독립적으로 동작하며 프로젝트 내부에 있는 파일 끼리는 묶어서 컴파일이 가능하다. C언어를 예로 들면 makefile과 같은 기능이다. |
| 코드 편집하기 | 사용자는 웹 브라우저를 통해 코드를 편집할 수 있다. |
| 컴파일 결과 확인하기 | 사용자는 웹 브라우저를 통해 컴파일 결과를 확인할 수 있다. 여기서 말하는 컴파일 결과 확인하기란, 컴파일 실패시 오류가 왜 발생했는지 알려주는 기능과 컴파일 성공시 실행 파일이 출력하는 결과를 포함한다. |
| 문제 출제하기 | 관리자는 별도의 문제 출제 페이지를 통해 문제를 출제할 수 있다. 이 때 문제의 정답을 정의하기 위하여 별도의 테스트 셋을 추가할 수 있다. |
| 문제 채점하기 | 사용자는 웹 브라우저에서 관리자가 출제한 문제를 코드로 구현하여 채점 받을 수 있다.  이 때 채점을 받는 기준은 관리자가 지정한 테스트 셋을 만족하는 결과가 몇개 인지에 따라 정답률 측정으로 평가된다. |
| VDB 관리하기 | 사용자는 별도의 VDB 페이지를 통해 자신의 VDB를 관리할 수 있다. 이 페이지에서는 자신이 코드에서 사용하고자 하는 database를 생성, 편집, 삭제할 수 있다. |
| 피드백 받기 | 사용자는 별도의 피드백 페이지를 통해 평소에 컴파일한 결과에 대한 피드백을 받을 수 있다. 피드백 결과에는 코드 스멜, 코드 중복도, 사용하지 않는 변수, 보안 취약성과 같은 정보가 포함된다. |

[Table 1. Main feature]

3.3 USE CASES

[Figure 2. T-HUB Use Case Diagram]

3.3.1 Use Case 1

|  |  |
| --- | --- |
| **Use Case 1** | |
| **Use Case Title** | Manage DB |
| **Primary Actor** | Novice Coder |
| **Text Description** | 1. Novice Coder가 Virtual database를 추가할 수 있다. 2. Novice Coder가 Virtual database를 변경할 수 있다. 3. Novice Coder가 Virtual database를 제거할 수 있다. |
| **Preconditions** | 1. Novice Coder는 로그인 상태이다. 2. Novice Coder는 project를 생성한 상태이다.   3. Novice Coder는 VDB를 생성한 상태이다. |
| **Basic Flow** | 1.1. Novice Coder가 VDB 메뉴를 클릭하여 VDB manager로 접속한다.  1.2. Novice Coder가 “New” 버튼을 누른다.  1.3. Novice Coder가 생성될 VDB 이름을 입력한다.  1.4. Virtual DB Manager가 VDB를 생성한다.  1.5. VDB 생성을 완료한다.  2.1. Novice Coder가 VDB 메뉴를 클릭하여 VDB manager로 접속한다.  2.2. Novice Coder가 편집하고 싶은 VDB를 선택한다.  2.3. Virtual DB Manager가 해당 VDB 정보를 제공한다.  2.4. Novice Coder가 편집하고자 하는 VDB 정보를 수정한다.  2.5. Virtual DB Manager가 수정된 정보를 VDB에 반영한다.  2.6. VDB 수정을 완료한다.  3.1. Novice Coder가 VDB 메뉴를 클릭하여 VDB manager로 접속한다.  3.2. Novice Coder가 제거하고 싶은 VDB를 선택한다.  3.3. Virtual DB Manager가 해당 VDB를 제거한다.  3.4. VDB 제거를 완료한다. |
| **Alternative Flows** | 1.4.a. 이미 존재하는 VDB 이름일 경우  1. 이미 존재하는 이름이라는 경고 메시지를 출력한다.  2.5.a. VDB 이름을 수정할 경우  1. Novice Coder로부터 VDB 이름을 입력받는다.  1-a. 이미 존재하는 VDB 이름일 경우  1. 이미 존재하는 이름이라는 경고 메시지를 출력한다.  2. Virtual DB Manager가 해당 이름으로 수정한다.  2.5.b. column을 추가할 경우  1. Novice Coder로부터 column 이름을 입력 받는다.  2. Virtual DB Manager가 해당 이름의 column을 추가한다.  2.5.c. column을 제거할 경우  1. Novice Coder가 제거할 column을 선택한다.  2. Virtual DB Manager가 해당 이름의 column을 제거한다. |
| **Post Conditions** | VDB를 추가, 변경, 제거했다. |

[Table 2. Use Case 1]

3.3.2 Use Case 2

|  |  |
| --- | --- |
| **Use Case 2** | |
| **Use Case Title** | Get Feedback |
| **Primary Actor** | Novice Coder |
| **Text Description** | 1. Novice Coder가 Feedback 결과를 불러올 수 있다. |
| **Preconditions** | 1. Novice Coder는 로그인 상태이다. 2. Novice Coder는 컴파일한 상태이다. |
| **Basic Flow** | 1.1. Novice Coder가 Feedback메뉴에 접속한다.  1.2. Novice Coder가 그간 컴파일한 목록 중에서 보고 싶은 피드백 정보를 선택한다.  1.3. 선택한 컴파일 정보에 맞추어 피드백 정보가 출력된다. |
| **Alternative Flows** |  |
| **Post Conditions** | Novice Coder가 Feedback정보를 받아보았다. |

[Table 3. Use Case 2]

3.3.3 Use Case 3

|  |  |
| --- | --- |
| **Use Case 3** | |
| **Use Case Title** | Make Code |
| **Primary Actor** | Novice Coder |
| **Text Description** | 1. Novice Coder가 파일을 추가할 수 있다. 2. Novice Coder가 코드를 변경할 수 있다. 3. Novice Coder가 파일을 제거할 수 있다. 4. Novice Coder가 코드를 컴파일 할 수 있다. 5. Novice Coder가 코드의 결과값을 확인할 수 있다. |
| **Preconditions** | 1. Novice Coder는 로그인 상태이다. 2. Novice Coder는 project를 생성한 상태이다. 3. Novice Coder는 파일을 생성한 상태이다. 4. Novice Coder가 작성한 코드가 정상적으로 컴파일 됐다. |
| **Basic Flow** | 1.1. Novice Coder가 Coding 메뉴를 클릭하여 Coding 페이지로 이동한다.  1.2. Novice Coder가 “New” 버튼을 클릭하여 파일을 생성한다.  1.3. Novice Coder가 생성하고자 하는 파일의 이름을 입력한다.  1.4. Code Manager 가 해당 파일을 생성한다.  1.5. Novice Coder의 웹 브라우저에 생성한 파일이 반영된다.  1.6. 파일 생성을 완료한다.  2.1. Novice Coder가 Coding 메뉴를 클릭하여 Coding 페이지로 이동한다.  2.2. Novice Coder가 “Edit” 버튼을 클릭하여 코드 파일을 선택한다.  2.3. Code Manager가 해당 코드 파일을 불러온다.  2.4. Novice Coder가 해당 코드를 수정한다.  2.5. Novice Coder가 “submit” 버튼을 누른다.  2.6. Code Manager가 수정 내용을 서버에 반영한다.  2.7. Novice Coder의 웹 브라우저에 수정된 코드 파일이 반영된다.  2.8. 코드 파일 수정을 완료한다.  3.1. Novice Coder가 Coding 메뉴를 클릭하여 Coding 페이지로 이동한다.  3.2. Novice Coder가 “Del” 버튼을 클릭하여 파일들을 선택한다.  3.3. Novice Coder가 “submit”버튼을 클릭한다.  3.4. Code Manager가 선택된 파일들을 서버에서 삭제한다.  3.5. Novice Coder의 웹 브라우저에 삭제된 파일이 반영된다.  3.6. 파일 삭제를 완료한다.  4.1. Novice Coder가 Coding 메뉴를 클릭하여 Coding 페이지로 이동한다.  4.2. Novice Coder가 “Compile” 버튼을 클릭한다.  4.3. Novice Coder의 웹 브라우저에 파일을 선택할 수 있는 메뉴가 나온다.  4.4. Novice Coder가 컴파일 하고자 하는 파일을 선택한다.  4.5. Novice Coder가 “submit”버튼을 클릭한다.  4.6. Code Manager가 선택된 파일들을 컴파일 한다. |
| **Alternative Flows** | 1.3.a. 이미 존재하는 파일 이름일 경우  1. 이미 존재하는 이름이라는 경고 메시지를 출력한다.  2.2.a. 서비스 할 수 없는 format일 경우  1. 서비스 할 수 없는 format이라는 오류 메시지를 출력한다.  4.6.a .선택된 파일이 단일 파일일 경우  1. 해당 파일만 컴파일 하여 결과를 반환한다.  4.6.b. 선택된 파일이 여러 개일 경우  1. 선택된 파일들을 묶어서 컴파일 하여 결과를 반환한다. |
| **Post Conditions** | 파일이 추가, 제거 되었다. 코드는 변경되었고, 컴파일, 결과를 확인했다. |

[Table 4. Use Case 3]

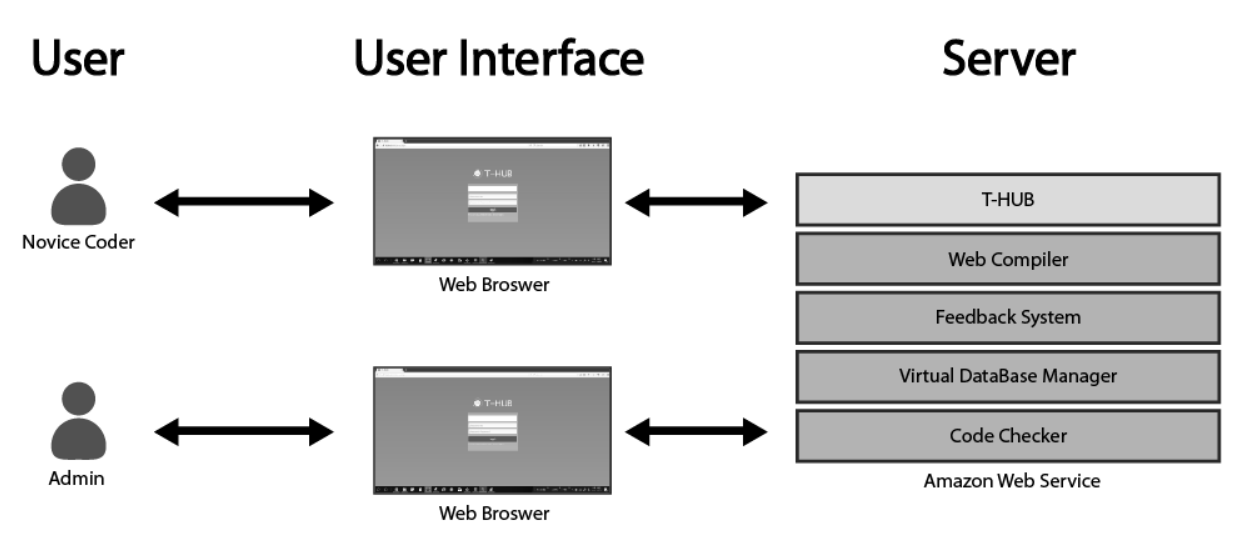
3.3.4 Use Case 4

|  |  |
| --- | --- |
| **Use Case 4** | |
| **Use Case Title** | Make Code Test |
| **Primary Actor** | Admin |
| **Text Description** | . 1. Admin은 문제를 추가할 수 있다.  2. Admin은 문제를 변경할 수 있다.  3. Admin은 문제를 삭제할 수 있다. |
| **Preconditions** | 1. Admin은 로그인 상태이다. 2. Admin은 문제를 생성한 상태이다. |
| **Basic Flow** | 1.1. Admin이 문제메뉴를 클릭한다.  1.2. Admin이 “New” 버튼을 누른다.  1.3. Admin이 생성할 문제 이름과 내용을 입력한다.  1.4. Admin이 “Save” 버튼을 누른다.  1.5. Code Checker 가 문제를 확인하고, 문제를 저장한다.  1.6. 문제 생성을 완료한다.  2.1. Admin이 문제메뉴를 클릭한다.  2.2. Admin이 “Edit” 버튼을 누른다.  2.3. Admin이 편집할 문제를 선택한다.  2.4. Code Checker 가 해당 문제를 제공한다.  2.5. Admin은 해당 문제를 편집한다.  2.6. Admin은 편집을 완료하고 “Save” 버튼을 누른다.  2.7. Code Checker 가 문제를 확인하고, 문제를 저장한다.  2.8. 문제 편집을 완료한다.  3.1. Admin이 문제메뉴를 클릭한다.  3.2. Admin이 “Del” 버튼을 누른 후, 삭제할 문제를 선택한다.  3.3. Admin은 “Submit” 버튼을 클릭한다.  3.4. Code Checker가 선택된 문제들을 서버에서 삭제한다.  3.5. 문제 삭제를 완료한다. |
| **Alternative Flows** | 1.3.a. 이미 존재하는 문제 이름일 경우  1. 이미 존재하는 이름이라는 경고 메시지를 출력한다.  1.5.a. 유효하지않은 문제인 경우  1. 에러 메시지를 출력한다.  2. 문제 내용 페이지로 돌아간다.  1.5.b. 유효한 문제인 경우  1. 문제를 저장한다.  2.7.a. 유효하지않은 문제인 경우  1. 에러 메시지를 출력한다.  2. 문제 내용 페이지로 돌아간다.  2.7.b. 유효한 문제인 경우  1. 문제를 저장한다. |
| **Post Conditions** | 문제가 추가,변경,삭제되었다. |

[Table 5. Use Case 4]

4 설계(DESIGN)

4.1 OVERALL 시스템 구조

[Figure 3. System Architecture]

4.1.1 User Interface

사용자는 웹 브라우저를 통해 T-HUB에 접근 한다. 사용자는 Firefox, Chrome, Micro Internet Explorer 등 여러 웹 브라우저를 사용 가능하다.

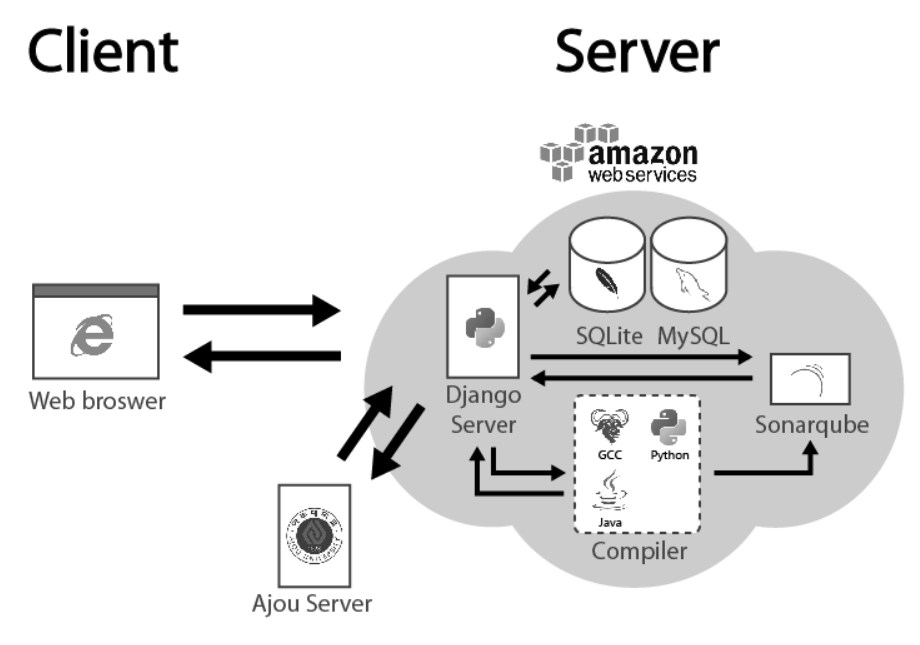
4.1.2 Server

T-HUB는 4개의 기능을 하는 layer로 구성되어있다. 각 layer가 하는 일은 아래와 같다.

|  |  |
| --- | --- |
| **Layer** | **function** |
| Web Compiler | 컴파일을 담당한다. 사용자가 컴파일 버튼을 누르면 현재 프로젝트 내에 있는 모든 코드들을 컴파일한다. |
| Feedback System | 피드백 기능을 담당한다. 사용자가 코드를 컴파일 했을 경우, 작성 한 코드의 퀄리티를 평가 해준다. |
| Virtual Database Manager | 사용자가 자유롭게 SQL문을 컴파일 할 수 있도록 사용자 정의 데이터베이스를 만들 수 있도록 도와준다. |
| Code Checker | 관리자가 문제를 만들때 문제를 옳게 만들었는지 체크하며, 사용자가 문제를 풀 때, 문제를 옳게 풀었는지를 체크해준다. 문제 채점 방식은 관리자가 문제를 만들 때 만든 여러 테스트 케이스의 정답을 가지고 사용자가 푼 답과 비교한다. |

[Table 6. System Layer]

4.2 SW ARCHITECTURE



[Figure 4. Software Architecture]

4.2.1 Client Side

사용자는 HTTP 프로토콜을 통해 서버와 통신을 한다. 보통은 GET메서드를 사용해 통신을 하며 로그인, 컴파일과 같은 기능의 경우는 POST메서드를 사용한다.

4.2.2 Server Side

T-HUB에서 대부분의 소프트웨어는 Server 쪽에 있다. 서버로는 Django 프레임워크를 사용하며, Django 프레임워크는 여러 소프트웨어를 활용하여 사용자에게 서비스를 제공한다. 데이터베이스로는 SQLite와 MySQL을 사용 한다. SQLite는 회원 정보나 문제 등과 같은 정보를 저장을 하고, MySQL의 경우에는 가상 데이터베이스에 필요한 정보 등을 저장한다. 컴파일은 GCC, Python, Java 컴파일러를 통해 진행된다. 피드백은 SonarQube를 이용해 진행이 된다. 로그인은 사용자의 편의성을 위해 아주대 포탈을 통해 학사 정보를 가지고 로그인을 하게 된다.

4.3 LIST OF COMPONENTS

|  |  |  |
| --- | --- | --- |
| **Type** | **Component** | **Description** |
| 기존 | SonarQube | 기존의 SonarQube가 제공하는 정보는 코딩 초심자가 보기에는 이해하기 힘든 정보들로 구성되어 있다. 우리 서비스는 이러한 정보를 초심자들도 보기 쉬운 형태로 가공함으로써 피드백 기능을 수행하도록 만든다. |
| 기존 | GCC compiler | GCC컴파일러를 통해 C언어 컴파일 기능을 제공한다. |
| 기존 | JAVA compiler | JAVA컴파일러를 통해 JAVA 컴파일 기능을 제공한다. |
| 기존 | PYTHON compiler | PYTHON컴파일러를 통해 PYTHON 컴파일 기능을 제공한다. |
| 기존 | jQuery | 기존의 자바스크립트는 가독성과 문법이 복잡하다 그래서 알아보기 쉬운 jQuery를 사용하여 가독성을 높인다. |
| 기존 | Django | 웹 IDE 플랫폼의 경우 문자열을 처리할일이 많다. PYTHON은 문자열 처리에 특화되어있고 이를 활용하는 Django고 웹 프레임워크를 사용한다. |
| 기존 | SQLite | 회원 정보, 코딩 문제와 같은 정보들을 저장하는 데이터 베이스이다. |
| 기존 | MySQL | 가상 데이터 베이스를 구현하기 위한 데이터베이스이다. |

[Table 7. List of components]

5 개발 관리

5.1 개발 환경 / 언어 / FRAMEWORK 활용 / 운영방안

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Client** | **Server** | **Tool** | **OS** | **Web Explorer** |
| - Javascript  - HTML5  - CSS | - Python 2.7  - Django | - SonarQube  - GCC  - Atom | - Linux Ubuntu 16.04 | - Firefox |

[Table 8. Development environment]

5.2 리스크 분석 및 회피 방안

|  |  |  |
| --- | --- | --- |
| **Risk Type** | **Risk** | **Solution** |
| Unskilled | SonarQube | 관련 자료 학습 및 실제 사용을 통해 해당 tool의 이해도를높이는데 주력하고, 만약 개발 방향과 맞지 않는다고 판단될 경우 자체적인 알고리즘을 구현하여 코드 피드백 기능을 구현하거나, 관련 Open source를 조사하여 활용할 수 있도록 한다. |
| Unskilled | Django | 관련 자료 학습 및 실제 개발을 통해 숙련도를 향상해 나간다.  만약 Django framework만으로 해결하기 힘든 문제를 만난다면, c, php와 같은 Django 이외의 언어를 통해 processing 하여 개발에 차질이 없도록 만든다. |
| System vulnerability | Compiler | 시스템에 위해를 가할 수 있는 위험 함수에 대한 통제가 필요하다.  실행 파일 형태의 코드에서는 시스템 계열의 함수와 입력 값의 buffer over flow를 집중적으로 체크하여 threat를 배제하고, 웹 형태의 언어의 경우 OWASP TOP10에서 제시하는 payload를 차단시켜서 threat를 제거한다. |
| System vulnerability | web system | 웹에서 발생할 수 있는 SQL Injection, XSS, URL Jumping, File upload attack과 같은 위협에 대비한 사전 처리가 필요하다.  첫 번째 방안으로 apache에서 자체적으로 제공하는 config file의 설정을 통해 root directory 제한 및 permission setting을 한다.  두 번째 방안으로 PYTHON 자체적으로 주요 input을 검사하는 함수를 만들어서, Injection에 잠재적으로 위험한 ‘. “, #와 같은 문자열을 필터링 한다. |

[Table 9. Risk Analysis and Avoidance Measures ]

5.3 개발 일정

|  |  |  |  |
| --- | --- | --- | --- |
| **Iteration** | **Features (Stories)** | **Task Description** | **Owner** |
| **1**  (10.10~10.23) | 개발 환경 구축 | SonarQube 설치 | 하재준 |
| 사용할 컴파일러 설치 | 이승환 |
| Django 설치 | 박진 |
| MySQL 설치 및 연동 | 엄준영 |
| Amazon Web Service 설정 | 이경재 |
| 패킷 분석 | 학사 로그인을 위한 패킷 분석 | 하재준 |
| 인트로 | 인트로 페이지 만들기 | 박진 |
| **2**  (10.24~11.06) | 학사 로그인 | 학사 정보를 이용한 로그인 기능 구현 | 하재준 |
| 로그인을 위한 데이터베이스 스키마 구현 | 엄준영 |
| 로그인 페이지 구현 | 박진 |
| 프로젝트 단위 관리 | 프로젝트 단위 관리 기능 구현 | 이승환 |
| 프로젝트 단위 관리를 위한 데이터베이스 스키마 구현 | 엄준영 |
| 코드(프로젝트 단위 관리를 포함한 상위) 페이지 구현 | 박진 |
| **3**  (11.07~11.20) | 피드백 기능 | 피드백 기능 구현 | 하재준 |
| 피드백 기능을 위한 데이터베이스 스키마 구현 | 엄준영 |
| 피드백 기능 페이지 구현 | 박진 |
| 문제 생성 | 문제 생성 기능 구현 | 이경재 |
| 문제 생성을 위한 데이터베이스 스키마 구현 | 엄준영 |
| 문제 생성 페이지 구현 | 박진 |
| **4**  (11.21~12.04) | 채점 시스템 | 채점 시스템 기능 구현 | 이경재 |
| VDB | VDB 설정 기능 구현 | 엄준영 |
| VDB 설정 페이지 구현 | 박진 |
| 컴파일러 기능 | 컴파일러 기능 구현 | 이승환 |
| 컴파일러 페이지 구현 | 박진 |
| **5**  (12.05~12.13) | 기능 별 테스트 및 구현 | 피드백, 로그인 기능 테스트 | 하재준 |
| 프로젝트 단위, 컴파일러 테스트 | 이승환 |
| 전체적인 웹 동작 테스트 | 박진 |
| 전체 데이터베이스, VDB기능 테스트 | 엄준영 |
| 문제 생성, 채점 시스템 테스트 | 이경재 |

[Table 10. Development Schedule]

5.4 비용 분석

|  |  |  |
| --- | --- | --- |
| **Subject** | **Description** | **Cost (Won)** |
| Amazon Web Service | - 1년 이상 사용자  - 데이터 전송량 15GB 초과  - 저장 공간 30GB 초과  위 조건에 해당 되지 않으므로 무료로 사용한다 | 0원 |
| 합계 | | 0원 |

[Table 11. Expense Analysis]

5.5 소스 코드 관리 및 CI 툴 운영 방안

[소스 코드 관리]

|  |  |
| --- | --- |
| **Tool** | **How to operate** |
| Git | Git을 통해 소스 코드를 관리 한다. 총 6개의 branch를 가질 것이며, 1개의 branch는 master branch로 본 프로젝트의 각각의 기능이 완성된 버전을 담고 있다. 나머지 5개의 branch는 각각 팀원들의 branch로 “dev+팀원의 이름 중 마지막 문자” 이름을 가지고 있다. 팀원들은 각자의 branch에서 맡은 기능들을 구현하며 각각의 기능들이 구현하면 그 때 마다 master에 merge하는 방식으로 Git을 사용할 예정이다. |

[Table 12. Source Code Management]

[CI 툴]

|  |  |
| --- | --- |
| **Tool** | **How to operate** |
| Jenkins | Jenkins을 사용해서 지속적으로 빌드를 한다. Git과 연동 할 것이며, 에러 발생시 에러를 낸 팀원에게 알리고 빠른 시간 내에 코드 수정본을Git의 master branch로 commit을 유도한다. |

[Table 13. Continuous Integration]

5.6 CODE CONVENTION

[ PYTHON ]

|  |  |
| --- | --- |
| **Type** | **Description** |
| Function | PYTHON 내의 주요 함수에는 함수 주석이 작성된다.  (example)  ‘’’  @ Author: ( 함수 작성자 )  @ Function: ( 함수 주요 기능 )  @ Date: ( 함수 제작일 )  @ Global variable: ( 함수 내에서 사용하는 전역 변수 )  ‘’’ |
| Variable | Global variable의 사용을 자제하도록 한다.  상황에 따라 Global variable의 사용이 필요하다면 한줄 주석을 통해 사용되는 함수명과 변수의 사용 목적을 되도록 상세히 기술한다.  (example)  FORBIDDEN = {“.pyc”, “sqlite3”} # used for Forbidden file extension  # called from foo1(), foo2() |
| Comment | 지나친 주석의 사용은 코드의 가독성을 떨어뜨리기 때문에 배제하도록 하며, 중추 역할을 하는 local variable과 주요 분기문에 한줄 주석 작성을 통해 그 역할을 분명하게 하도록 한다. |

[Table 14. Code Convention of PYTHON]

6 업무 분장 계획

6.1 FEATURE / 컴포넌트 기반

|  |  |
| --- | --- |
| **Member** | **Division Of Works** |
| 하재준 | * 학사 로그인 기능 구현 * 피드백 기능 구현 |
| 이승환 | * 프로젝트 단위 관리 구현 * 웹 컴파일러 구현 |
| 박진 | * 웹 페이지 구현 * 디자인 |
| 엄준영 | * 데이터베이스 스키마 설계 * VDB 기능 구현 |
| 이경재 | * 문제 생성 구현 * 채점 시스템 구현 |

[Table 15. Division of works]

6.2 기타과제관리/DEPOLY/발표 등 부가적 업무 분장

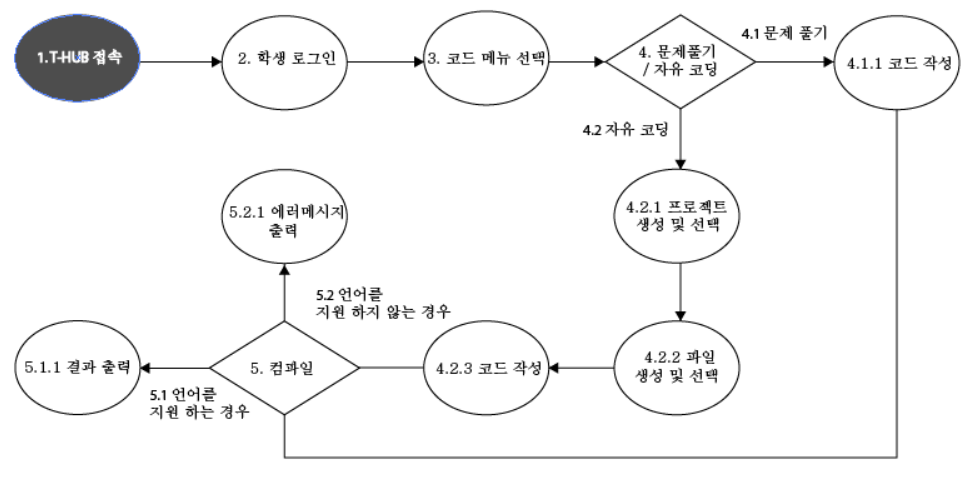
|  |  |
| --- | --- |
| **Member** | **Additional Division Of Works** |
| 하재준 | 전체적인 일정 관리 |
| 이승환 | 기타 공모전 참가 신청 |
| 박진 | DEPLOY 계획 작성 |
| 엄준영 | 산학협력 프로젝트 수행 계획서 작성 |
| 이경재 | 컨셉, 기획, 중간, 최종 발표 |

[Table 16. Additional division of works]

7 DEPLOY 계획(DEMO)

7.1 데모 시나리오

7.1.1 데모 시나리오 1

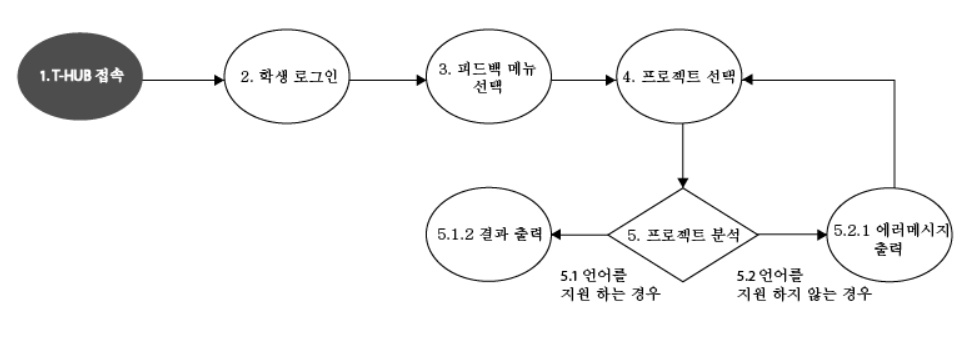


[Figure 5. Demo scenario 1 flowchart]

|  |  |
| --- | --- |
| **Demo Scenario 1** | |
| **Title** | 코드 컴파일 |
| **Actor** | User |
| **데모 시나리오** | 1. T-HUB 웹 페이지에 접속한다.  2. 학생 아이디로 로그인한다.  3. 코딩 메뉴를 선택한다.  4. 문제 풀기와 자유 코딩 중 하나를 선택한다.  4.1 문제 풀기의 경우  4.1.1 코드를 작성한다.  4.2 자유 코딩의 경우  4.2.1 프로젝트를 생성(혹은 선택)한다.  4.2.2 파일을 생성(혹은 선택)한다.  4.2.3 코드를 작성한다.  5. 컴파일한다.  5.1 지원이 되는 언어의 경우  5.1.1 컴파일 결과를 출력한다.  5.2 지원이 되지 않는 언어의 경우  5.2.1 에러메시지 출력  5.2.2 4번으로 돌아간다. |

[Table 17. Demo scenario 1]

7.1.2 데모 시나리오 2

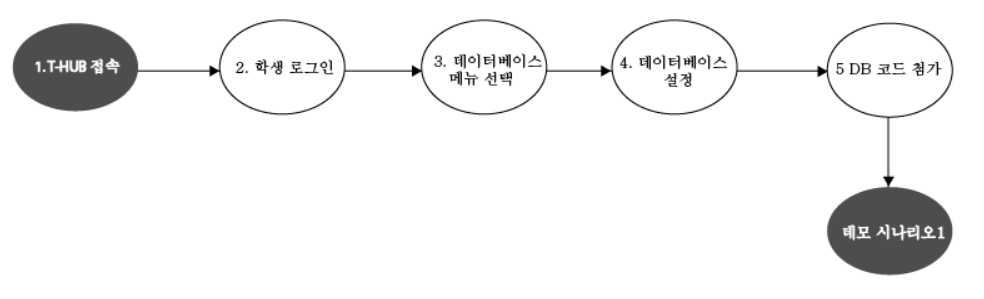


[Figure 6. Demo scenario 2 flowchart]

|  |  |
| --- | --- |
| **Demo Scenario 2** | |
| **Title** | 피드백 기능 |
| **Actor** | User |
| **데모 시나리오** | 1. T-HUB 웹 페이지에 접속한다.  2. 학생 아이디로 로그인한다  3. 피드백 메뉴를 선택한다.  4. 결과를 확인할 프로젝트를 선택한다.  5. 선택한 프로젝트를 분석해준다.  5.1 지원이 되는 언어의 경우  5.1.1 피드백 결과를 출력한다.  5.2 지원이 되지 않는 언어의 경우  5.2.1 에러메시지 출력  5.2.2 4번으로 돌아간다. |

[Table 18. Demo scenario 2]

7.1.3 데모 시나리오 3

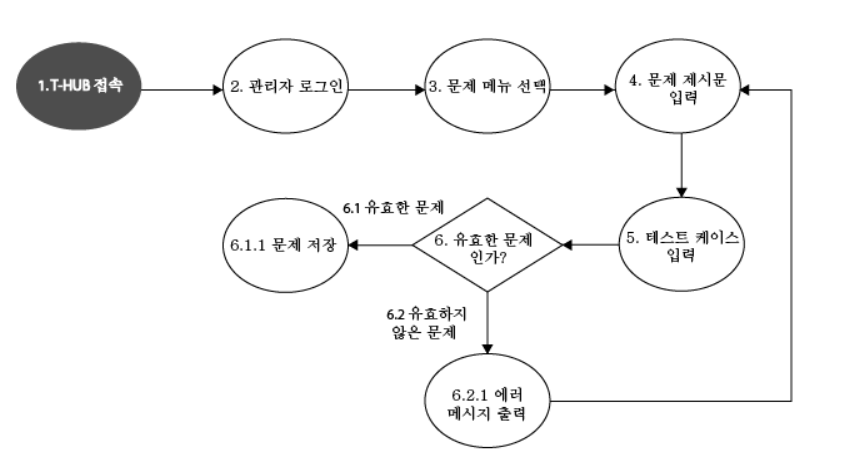


[Figure 7. Demo scenario 3 flowchart]

|  |  |
| --- | --- |
| **Demo Scenario 3** | |
| **Title** | 가상 데이터베이스 |
| **Actor** | User |
| **데모 시나리오** | 1. T-HUB 웹 페이지에 접속한다.  2. 학생 아이디로 로그인한다  3. 데이터베이스 메뉴를 선택한다.  4. 사용자에 맞는 데이터베이스 설정을 한다.  5. 코드에 DB구문을 첨가하여 **데모 시나리오 1**의 과정을 반복한다. |

[Table 19. Demo scenario 3]

7.1.4 데모 시나리오 4



[Figure 8. Demo scenario 4 flowchart]

|  |  |
| --- | --- |
| **Demo Scenario 4** | |
| **Title** | 문제 생성 |
| **Actor** | User |
| **데모 시나리오** | 1. T-HUB 웹 페이지에 접속한다.  2. 관리자 아이디로 로그인한다  3. 문제 메뉴를 선택한다.  4. 문제 제시문을 입력한다.  5. 테스트 케이스들을 입력한다.  6. 문제가 유효한지 확인한다.  6.1 유효한 문제  6.1.1 문제를 저장한다.  6.2 유효하지 않은 문제  6.2.1 에러메시지 출력  6.2.2 4번으로 돌아간다. |

[Table 20. Demo scenario 4]

7.2 KPI(KEY PERFORMANCE INDICATOR)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Result Area** | **Outcome Indicator** | **Goal** | **Ratio(%)** | **Target** |
| 빠른 결과 출력 | 컴파일 속도 | 3s 이내 | 30% | 사용자 |
| 피드백의 정확도 | 고객의 만족도 | 50%이상 | 30% | 사용자 |
| 일정 준수 | 일정 달성률 | 80%이상 | 40% | 개발자 |

[Table 21. KPI]

7.3 FUTURE PLAN

|  |  |
| --- | --- |
| **Goal** | **Description** |
| 컴파일 및 피드백 여러 언어 추가 | C, JAVA, PYTHON 언어 뿐만 아니라 다양한 언어 를 제공해줌으로써, 사용자에게 편의를 제공한다. |
| 다양한 문제 추가 | 기본예제 뿐만아니라 다양한 알고리즘 문제를 추가하여 실질적인 교육에도 도움이 될 수 있도록 한다. |
| 다양한 외부 로그인 포털 사이트 증대 | 다양한 로그인 API를 추가하여, 아주대학교 학생뿐만아니라 모든 사람들이 사용할 수 있도록 확장성을 추가한다. |

[Table 22. Future plan]

1. https://www.weforum.org/agenda/2016/01/the-fourth-industrial-revolution-what-it-means-and-how-to-respond/ [↑](#footnote-ref-1)
2. <http://news1.kr/articles/?3123013> [↑](#footnote-ref-2)
3. <http://magazine.hankyung.com/business/apps/news?popup=0&nid=01&c1=1013&nkey=2017070301127000431&mode=sub_view> [↑](#footnote-ref-3)