Software Engineering Team CtrlZ

과제 제출 시스템

|  |  |  |
| --- | --- | --- |
| 담당 교수님 |  | 최윤자 교수님 |
| 팀장 |  | 유동규 2011105062 |
| 팀원 |  | 박동원 2011105038 |
| 팀원 |  | 윤준호 2011105065 |
| 팀원 |  | 이승진 2011105072 |

**INDEX**

1. Overview
2. Requirements
3. Principal Contents
4. Task Plan and Distribution
5. Strategies
6. Special
7. **Overview**

과제 제출 시스템은 기존 ABEEK사이트를 이용한 과제 제출시 부족했던 보안 및 데이터 관리 등을 보완하기 위해 계획 및 제작되었다. 그러나 다음과 같은 개선사항들이 발견되었다.

1. 현재까지 개발 된 과제 제출 시스템은 네트워크 연동과 보안에 관한 기능이 구현되어있지 않다.
2. 과제 게시물의 등록 및 채점 시에 사용자들이 그 사실을 확인할 수 있는 알림 기능이 필요하다.
3. 게시물의 수정 및 삭제 기능이 존재하지 않는 데 따른 불편함이 있다.
4. 설계 측면에 있어 MVC 패턴을 적용하여 유지보수 측면에서의 용이성을 끌어올릴 필요가 있다.

이번 프로젝트는 위에서 언급된 부분들의 추가 구현 및 개선에 대한 계획이 포함되어있다.

1. **Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **기존**  **요구사항** |  | 1. 교수, 조교, 학생이 시스템을 사용한다. 2. 교수, 조교는 과제를 등록한다. 3. 학생은 교수, 조교가 올린 과제를 확인하고, 제출한다. 4. 교수, 조교는 학생이 제출한 과제를 확인하고 평가 및 코멘트를 단다. 5. 학생은 교수, 조교에게 질문이 가능하다. 6. 교수, 조교는 학생의 질문에 답변이 가능하다. 7. 학생은 자기 자신의 과제 및 채점 결과만 볼 수 있다. |
| **추가**  **요구사항** |  | 1. 교수, 조교가 과제를 올리거나, 평가를 하면 해당 학생에게 알림이 간다. 2. 학생이 과제를 제출하면 교수, 조교에게 알림이 간다. 3. 네트워크로 모두 연결되어 있어야 한다. 4. 게시물 보안 및 통신 데이터의 암호화 처리를 한다. |
|  | **주요 기능 :** 과제 등록, 과제 제출, 과제 평가, 질문 답변, 보안, 알림   * **과제 등록 (2) :** 교수, 조교는 자신이 맡은 과목 게시판에 새로운 과제를 등록할 수 있다. * **과제 제출 (3) :** 학생은 자신이 듣는 과목의 과제 게시물에 자신이 작성한 과제를 제출 할 수 있다. * **과제 평가 (4) :** 교수, 조교는 학생이 제출한 과제를 확인하고 점수를 매기거나 코멘트를 달아 줄 수 있다. * **질문 답변 (5), (6) :** 학생은 과목의 질문 게시판에 질문을 올릴 수 있고, 교수, 조교, 학생은 질문에 답변을 해줄 수 있다. * **알림 (8), (9) :** 기존 Abeek 등의 과제 제출 시스템은 알림 시스템이 없어 수시로 확인했어야 했는데, 이를 개선해서 과제 등록, 제출, 평가 시 알림이 가게 한다. * **보안 (7), (11) :** 기존 과제 제출 시스템의 보안 문제를 해결하기 위해, 데이터는 암호화하고, 학생은 자기 자신의 과제와 평가, 모든 과제의 평균 점수만 알 수 있다. | | |

1. **Principal Contents**

|  |  |  |
| --- | --- | --- |
| **개발** |  | 1. 네트워크  * 로그인한 모든 사용자들은 서버에 접속해서 특정 데이터에 접근 가능   + 교수, 조교는 자신이 강의하는 과목의 과제 게시판, 질문 게시판, 학생들 과제, 제출한 과제, 평가된 과제 데이터에 접근 가능   + 학생은 자신이 수강하는 과목의 과제 게시판, 질문 게시판, 자신의 과제, 자신의 평가된 과제, 해당 과제의 전체 평균 데이터에 접근 가능하다. * 사용자들의 계정 정보는 서버에 저장됨 * 게시물 데이터는 서버에 저장됨  1. 보안  * 통신 되는 모든 데이터는 암호화 처리 됨 * 서버에 데이터가 저장될 때 암호화를 거침 * 클라이언트가 데이터를 요청하면 서버가 판별 후 복호화 해서 클라이언트에게 데이터 전송 * 학생은 자신의 과제와 결과, 또한 전체 평균만을 확인할 수 있음  1. 알림  * 학생이 과제 제출 시 교수, 조교에게 과제 제목, 학생 이름, 제출 시간이 있는 알림이 감 * 학생의 과제가 성공적으로 제출 되었을 경우 과제 제목, 제출 시간이 있는 알림이 본인에게 감 * 교수, 조교가 과제를 채점할 시 학생에게 과제 이름, 채점 시간, 점수가 있는 알림이 감 * 알림이 가야 할 상대가 접속해있지 않을 경우 서버에 알림 데이터를 저장해놓고 추후 접속 시 알림이 가도록 함 |
| **개선** |  | 1. MVC 패턴 적용  * Class 들을 Model, View, Controller 세 가지 요소로 분리함  1. 글 수정 및 삭제 기능 추가  * 학생이 올린 과제 게시물의 수정 및 삭제 기능 추가 * 교수, 조교가 등록한 과제에 대한 수정 및 삭제 기능 추가 * 교수, 조교의 과제 평가에 대한 수정 및 삭제 기능 추가 * 질문 게시물의 수정 및 삭제 기능 추가 * 댓글의 수정 및 삭제 기능 추가 |
| **역점 사항**  MVC 패턴 적용, 네트워크, 보안, 알림, 글 수정 및 삭제 기능  **우선 순위**  MVC 패턴 적용 -> 네트워크 연동 -> 보안 시스템 적용 -> 글 수정 및 삭제 기능 -> 알림 기능  **목표**  위의 우선 순위를 따라 개발을 진행 예정  최소 네트워크, 보안은 완성할 예정 | | |

**4. Task Plan and Distribution**

**Task Priority**

1. MVC 패턴 적용
2. 네트워크 연동
3. 보안 시스템 적용
4. 수정 및 삭제 기능 추가
5. 알림 기능 추가

**Task Distribution**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 유동규 |  | 박동원 |  | 윤준호 |  | 이승진 |
| 네트워크  보안 |  | 네트워크  MVC 패턴 적용 |  | 알림  MVC 패턴 적용 |  | 수정 및 삭제  알림 |

**Member Allocation Chart**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 업무 분담 | | | |
| 유동규 | 보안 | | 네트워크 | |
| 박동원 | MVC 패턴 적용 | 네트워크 | | |
| 윤준호 | MVC 패턴 적용 | | 알림 | |
| 이승진 | 수정 및 삭제 | | | 알림 |

**Schedule**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 4.3 | 4.4 | 4.5 | 5.2 | 5.3 | 5.4 | 5.5 | 6.1 | 6.2 | 6.3 |
| 계획 수립 |  |  |  |  |  |  |  |  |  |  |
| 디자인 분석 |  |  |  |  |  |  |  |  |  |  |
| MVC 패턴 적용 |  |  |  |  |  |  |  |  |  |  |
| Test Code 작성 및 테스트 |  |  |  |  |  |  |  |  |  |  |
| 실제 기능 개발 |  |  |  |  |  |  |  |  |  |  |
| 테스트 분석, 코드 품질 향상 |  |  |  |  |  |  |  |  |  |  |
| 최종 발표 준비 |  |  |  |  |  |  |  |  |  |  |

**5. Strategies**

|  |  |  |  |
| --- | --- | --- | --- |
|  | MVC 패턴 공부 |  | 시스템 개선사항으로서 MVC 패턴 적용이 있으므로 MVC패턴의 형태나 적용법 등에 대해 확실히 알아두어야 한다. |
| 자바 네트워크 공부 |  | 네트워크를 기반으로 하는 서버와의 통신이 가능토록 시스템을 개발하기 위해 자바 네트워크 프로그래밍에 숙달되어야 한다. 서버, 클라이언트, 소켓 프로그래밍에 대해 서로 알아보고 공부하도록 한다. |
| 기존 시스템의 이해 |  | 기존 시스템을 깊이 이해하는 것이 개선은 물론 기능의 추가에 있어서도 반드시 필요하다. 먼저 기존 시스템의 문서 정독, 클래스 다이어그램 분석, 실제 프로그램 구성 분석 등을 한다. |
| 정확한 업무분담 및 확인 |  | 각자의 능력과 장점에 맞는 업무를 분담하고 진행 정도를 매주 확인하여 효율과 진행속도를 높인다. |
| 요구사항 재분석 및 테스트 |  | 요구사항의 재분석과 테스트를 통해 잘못 파악했거나 수정된 사항을 즉각 반영할 수 있어야 한다. |
| GIT을 이용한 버전 관리 및 분업 |  | 버전 관리 툴인 GIT을 이용하여 실제 프로젝트에서 버전 관리 및 백업을 하고, GIT 사용법을 익혀 팀원간 분업을 한다. |
| TDD를 활용한 개발 |  | 테스트 코드를 작성하여 원하는 부분의 완성 여부를 빠르게 파악하고 신뢰도를 높일 수 있어야 한다. |

**6. Special**

UI 단순히 프로그램 개발에서 끝나는게 아닌 실제 사용자가 사용하기 편하게 UI 개발