산학프로젝트 제안서

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 과목명 | 산학프로젝트 | | 팀명 | | Five guys | 조 번호 | | | 2 |
| 프로젝트명 | MBTIng | | | | | | | | |
| 구성원 및 역할 | 이름 | 황현택 | | 이메일 | | | | hhtboy1024@gmail.com | |
| 번호 | 01026558193 | | | | | | | |
| 역할 | 팀 리더, 백엔드 개발 | | | | | | | |
| 이름 | 이영진 | | 이메일 | | | | leesu0229@naver.com | |
| 번호 | 01066940678 | | | | | | | |
| 역할 | 팀 부리더, 프론트엔드 개발 | | | | | | | |
| 이름 | 장혁수 | | 이메일 | | | | astg98@gmail.com | |
| 번호 | 01085674138 | | | | | | | |
| 역할 | 서기, 프론트엔드 개발 | | | | | | | |
| 이름 | 최율 | | 이메일 | | | | choiyool00@gmail.com | |
| 번호 | 01090126399 | | | | | | | |
| 역할 | 서기, 프론트엔드 개발 | | | | | | | |
| 이름 | 홍은기 | | 이메일 | | | | ghddmsrl1@naver.com | |
| 번호 | 01058500159 | | | | | | | |
| 역할 | 서기, 백엔드 개발 | | | | | | | |
| 프로젝트 제안 및 요구사항 | 주제 : 대학생을 위한 MBTI 기반 블라인드 소개팅 웹 애플리케이션.  프로젝트의 세 가지 키워드 <MBTI, Blind, chatGPT>   1. MBTI : MBTI기반 상대 매칭 알고리즘을 적용. 2. Blind : 상대에게 프로필 사진이 공개되지 않고, 채팅 기능을 통해 많은 대화를 유도. 3. chatGPT : 매칭 이후 데이트에 도움이 될 정보를 인공지능이 제공(ex: 데이트 코스 추천)   대부분의 소개팅 앱을 살펴보면 프로필 사진을 보고 상대를 결정하게 됩니다. 이러한 방법으로 소개팅을 진행하면 서로에 대해 아는 정보도 적을 뿐더러, 얼굴이 노출되기 때문에 신상 정보 노출의 위험이 있거나 신상 도용 등의 문제가 있습니다.  따라서 저희는 프로필 사진을 제거해 블라인드 형식의 소개팅 서비스를 제공하고, MBTI 기반의 매칭 시스템을 사용하여 더 잘 맞는 상대를 만날 수 있는 기회를 제공합니다. 또한 대학교 인증 api와 카카오톡 본인인증 api를 사용해서 서로에 대한 신뢰도가 올라갑니다. 마음에 드는 상대를 찾게 되면 상대방에게 호감 표시를 보낼 수 있고, 상대방이 호감으로 동의를 해주면, 채팅을 통해 상대방과 대화를 진행해 볼 수 있습니다. 충분한 대화를 진행한 후, 결제를 통해 추가로 대화를 진행하거나 상대방의 카카오톡ID 또는 연락처를 받아볼 수 있습니다. 매칭이 성사된 이후는 최신 인공지능인 ChatGPT의 데이트 코스를 추천 받는 등의 도움을 받아볼 수 있습니다.  <기술적 요구사항>   1. 회원가입 (DB 구축) 2. 대학 / 본인 인증 3. 마이 페이지 (호감 로그, 채팅방) 4. 회원 매칭(MBTI 기반 알고리즘) 5. 소켓을 활용한 채팅 기능 6. 가상 결제 시스템 7. chatGPT api 8. AWS 클라우드 서비스 환경 구축 | | | | | | | | |
| 프로젝트  기대효과 | 이 프로젝트는 대부분의 소개팅 서비스 시장처럼 가벼운 마음을 가지고 하는 것이 아닌, 진지한 만남을 추구합니다. 신뢰를 주는 본인인증 방식과, 프로필 사진이 비공개라는 점에서, 보다 더 진솔한 대화를 나눠 서로에 대해 알아갈 수 있습니다. 타겟층을 대학생으로 제한하고, 젊은 세대들이 좋아하는 주제인 MBTI를 활용함으로써 많은 고객층을 확보할 수 있습니다.  추가적으로 chat gpt의 데이트 코스 추천 등 최신 ai의 기능을 이용해서 양질의 대답을 받을 수 있습니다. | | | | | | | | |
| 개발 내용 및  필요 기술 | 크게 프론트 엔드와 백 엔드로 나눠서 개발을 하게 됩니다.  <프레임워크>  프론트 엔드 : React  백 엔드 : Spring boot  <프로세스 모델>  Linear Process Flow(테스트 위주 waterfall)  시간이 정해져 있고 기획을 고정해두고 개발을 진행하기 때문.  <필요 기술>  웹 서버 : AWS 클라우드 서비스를 이용하여 서버와 데이터베이스를 올립니다.  프론트 엔드는 반응형 설계와 CRUD 설계가 필요합니다.  외부 api : chatGPT, sms 본인인증  소켓을 활용하여 채팅 기능  <마일스톤 및 일정 관리>  Notion을 활용한 회의록  Zoom을 이용한 비대면 회의 | | | | | | | | |
| 구성원 별 상세 개발내용 및 R&R | 프론트엔드의 경우, 4월 달에 UI설계가 나와야 정확한 분업이 가능.  백엔드는 기본적인 인터페이스는 같이 설계하고, 구현은 메소드 별 나눠서 분업.  <프론트 엔드 - 이영진>  - 프론트엔드 개발 환경 설정 및 패키지 관리  - 프로젝트 컨벤션 설계 및 조율  - 프론트엔드 일정과 이슈 관리 및 분배  - UI/UX 디자인 | | | | | | | | |
| <프론트 엔드 - 장혁수>  - 서기(회의록 및 보고서 작성)  - 백엔드와 협업하면서 API 핸들링을 통해 서버와 데이터 주고 받기  - 테스팅 및 디버깅 | | | | | | | | |
| <프론트 엔드 - 최율>  - 서기(회의록 및 보고서 작성)  - 프로젝트 아키텍처 설계  - 재사용이 가능한 형태로 디자인과 CSS를 같이하면서 컴포넌트화 | | | | | | | | |
| <백 엔드 - 황현택>  - 매주 회의록 초안 작성  - 노션 관리(일정 관리, 카탈로그 나누기, 기획서 작성 등)  - AWS 관리  - 클래스 다이어그램 설계  - 대학생 인증 api 인터페이스 구현  - 이후 메소드 별로 담당해서 개발 | | | | | | | | |
| <백 엔드 - 홍은기>  - 서기(회의록 및 보고서 작성)  - mvc 아키텍처 설계 - 내부 비즈니스 로직 인터페이스 구현  - 이후 메소드 별로 담당해서 개발 | | | | | | | | |
| Technical Architecture | 프레임워크 : React(프론트엔드) / Spring Boot(백엔드)  데이터베이스 : MariaDB  클라이언트 서버 통신 방식 : REST api  클라우드 서비스 : AWS lightsail  Api 명세 : swagger  외부 api : chatGPT, sms 본인인증  협업 방식 : Zoom, Github, Notion  다이어그램 설계 툴 : draw.io | | | | | | | | |
| 개발에 발생할 수 있는 Risks  (예상 문제점 및 대책) | <프론트 엔드 관점>   1. CORS 이슈 다른 출처의 리소스를 사용할 때 발생하며, CORS가 무엇이고 어떤 이슈를 낳는지 미리 공부하여 예방합니다. 2. 로그인 및 회원가입시 보안 이슈 사용자의 민감한 정보를 브라우저에 그대로 노출되지 않게, 비밀번호 필드 암호화 및 HTTPS, SSL을 사용합니다. 3. 브라우저 호환 이슈 브라우저별로 호환되는 HTML태그나 JS문법이 다르기 때문에 해당 사항을 고려합니다. | | | | | | | | |
| <백 엔드 관점>   1. 기획의 수정으로 클라이언트가 요청하는 api 인터페이스가 변경됨 SOLID원칙과 DI를 적용시켜 변화에 유연하게 설계를 하고, 부족한 부분은 명세를 확실하게 하여 변동이 없도록 합니다. 2. 개인정보 노출 위험(XSS, sql injection) spring의 보안 솔류션을 최대한 적용하고, 추가로 필요한 부분은 필터링 코드를 작성하는 등으로 해결합니다. 3. 트래픽 초과(chatGPT의 부적절한 답변) api 결과를 일정 형식으로 받게 하여 해결합니다. | | | | | | | | |
| <비용 발생>   * Aws lightsail 요금제 * chatGPT api 요금제 * 본인인증 api 요금제 * 도메인 구매 * 노션 요금제 | | | | | | | | |
| 주요일정 | <공통>  제안서 발표 전까지 기획서 수정, swagger을 통한 api 명세 작성  <프론트엔드>  4월 전까지 피그마를 통한 UI작업  4월부터 본격 개발 착수, 페이지 라우팅 작업 우선 진행, 이후 CSS 작업  <백엔드>  4월 전까지 aws 서버와 데이터베이스 올리는 작업, 스프링 부트 기본 아키텍쳐 설계  4월부터 본격 개발 착수, 대학생 인증 api, 매칭 비즈니스 로직, 채팅 기능 우선 개발  이후 chat gpt api와 결제기능 등 추가 기술 구현 | | | | | | | | |
| 상세 개발일정 | 프론트엔드 상세 일정(보다 상세한 일정은 4월 초 UI설계 완료 후 추가 예정) | | | | | | | | |
| 피그마로 UI 설계 | | | | | | 3월 말 까지 | | |
| 백엔드와 함께 DB설계 | | | | | | 3월 말 ~ 4월 초 까지 | | |
| 화면구성에 필요한 Components 및 Routing 설계 | | | | | | 4월 1주차 | | |
| React로 개발(Dummy data 활용) | | | | | | 4월 2주차 ~ 5월 2주차 | | |
| 백엔드 단에서 서버 개발되면 실제로 api 호출하며 개발 | | | | | | 5월 3주차 | | |
| CSS 스타일링 | | | | | | 5월 3주차 | | |
| 테스트 및 디버깅 | | | | | | 5월 4주차 | | |
| 성능 개선 | | | | | | 프로젝트 종료 까지 | | |
| 백엔드 상세 일정 | | | | | | | | |
| 클래스 다이어그램, db 설계 | | | | | | 3/26까지 | | |
| 패키지 & 클래스 생성 | | | | | | 4/2까지 | | |
| mvc 패턴 적용 | | | | | | 4/9까지 | | |
| 회원가입 시스템 | | | | | | 4/16까지 | | |
| 본인 인증 | | | | | | 4/23까지 | | |
| 대학생 인증 | | | | | | 4/23까지 | | |
| mariaDB 연동 | | | | | | 프로젝트 전반 연동 | | |
| 내부 매칭 비즈니스 로직 | | | | | | 4/30까지 | | |
| 채팅 기능 구현 | | | | | | 5/14까지 | | |
| 가상 결제 시스템 | | | | | | 5/21까지 | | |
| chatGPT api | | | | | | 5/28까지 | | |
| 테스트 및 디버깅 | | | | | | 5/31까지 | | |