Software 프로젝트 프로세스

1. **요구사항 분석 (Requirements Analysis)**
    - 무엇을 만들 것인가
2. **설계 (Design)**
    - 어떻게 만들 것인가
3. **구현 (Implementation)**
    - 만들기
4. **테스트 (Testing)**
    - 잘 만들었는지 확인
5. **배포 (Deployment)**
    - 잘만들었으면 사용
6. **유지보수 (Maintenance)**
    - 사용하다 보면 업그레이드가 필요. 기능 추가, 기능 제거

요구사항 분석

1. 요구사항 분석의 정의

- 요구사항 분석(Requirements Analysis)은 소프트웨어 개발 프로젝트의 초기 단계에서 **사용자와 이해관계자의 요구를 수집하고, 이를 명확히 정의하고 문서화하는 과정**이다.

2. 요구사항 분석의 목적

- 사용자의 기대와 필요를 정확히 이해하고 이를 소프트웨어 시스템에 반영하여, 최종 제품이 기대에 부응할 수 있도록 한다.
- 개발 과정에서 발생할 수 있는 혼란과 변경을 최소화하고, 프로젝트의 효율성을 높인다.

3. 요구사항 분석 주요 내용

- **요구사항 수집**: 다양한 방법(인터뷰, 설문조사, 워크샵 등)을 통해 사용자와 이해관계자의 요구를 수집한다.
- **요구사항 명세**: 수집된 요구사항을 명확하고 구체적으로 문서화한다.
- **요구사항 검증**: 수집된 요구사항이 일관성 있고, 완전하며, 실현 가능한지 검토한다.
- **요구사항 관리**: 요구사항의 변경을 관리하고 추적하며, 프로젝트 전체에 걸쳐 요구사항이 반영될 수 있도록 지속적으로 관리한다.

4. 산출물

- **요구사항 명세서(SRS)**: 시스템의 기능적 요구사항과 비기능적 요구사항을 상세히 기술한 문서.
- **요구사항 추적 매트릭스(RTM)**: 요구사항의 추적성을 보장하여, 요구사항이 설계, 구현, 테스트 단계에서 잘 반영될 수 있도록 하는 도구.

5. 주요 요구사항 수집 기법

- 요구사항 수집은 사용자와 이해관계자의 요구를 체계적으로 모으는 과정이다.
- 다양한 기법을 통해 사용자와의 의사소통을 강화하고, 명확하고 구체적인 요구사항을 도출하는 것이 목적이다.

1. **인터뷰 (Interviews)**:
    - **정의**: 사용자와 일대일 대화를 통해 요구사항을 직접 수집하는 방법.
    - **장점**: 깊이 있는 정보를 얻을 수 있고, 사용자 요구를 명확히 이해할 수 있다.
    - **단점**: 시간과 비용이 많이 들 수 있으며, 인터뷰어의 기술에 따라 결과가 달라질 수 있다.
2. **설문조사 (Surveys/Questionnaires)**:
    - **정의**: 다수의 사용자에게 일련의 질문을 작성하여 요구사항을 수집하는 방법.
    - **장점**: 많은 사람들에게 빠르게 정보를 수집할 수 있다.
    - **단점**: 응답률이 낮을 수 있으며, 깊이 있는 정보를 얻기 어렵다.

3. **워크샵 (Workshops)**:
    - **정의**: 다양한 이해관계자가 모여 집단 토론을 통해 요구사항을 도출하는 방법.
    - **장점**: 다양한 관점을 한 번에 수집할 수 있으며, 협력과 합의를 이끌어내기 좋다.
    - **단점**: 많은 시간과 조정이 필요하며, 모든 참가자의 참여를 이끌어내기 어려울 수 있다.
4. **관찰 (Observation)**:
    - **정의**: 실제 업무 환경에서 사용자의 작업을 관찰해 요구사항을 파악하는 방법.
    - **장점**: 사용자가 말로 표현하지 못하는 요구사항을 발견할 수 있다.
    - **단점**: 관찰자의 주관이 개입될 수 있으며, 시간이 많이 소요될 수 있다.

5. **포커스 그룹 (Focus Groups)**:
    - **정의**: 특정 주제에 대해 다양한 배경을 가진 사용자들이 모여 토론하는 방법.
    - **장점**: 다양한 아이디어와 요구사항을 한 번에 수집할 수 있다.
    - **단점**: 그룹 성격에 따라 결과가 영향을 받을 수 있으며, 일부 의견이 과도하게 반영될 수 있다.
6. **브레인스토밍 (Brainstorming)**:
    - **정의**: 자유로운 아이디어 발상을 통해 요구사항을 도출하는 방법.
    - **장점**: 창의적인 아이디어를 많이 얻을 수 있으며, 팀의 참여를 유도할 수 있다.
    - **단점**: 실현 가능성이 낮은 아이디어가 많이 나올 수 있으며, 토론이 길어질 수 있다.

수집 기법 선택 시 고려사항

- **프로젝트의 규모와 복잡성**: 대규모 프로젝트일수록 다양한 기법을 병행하는 것이 좋다.
- **이해관계자의 접근성**: 이해관계자와의 접근성이 높을수록 직접적인 기법(인터뷰, 워크샵)을 활용할 수 있다.
- **예산과 시간**: 각 기법의 비용과 시간 소요를 고려하여 선택한다.

요구사항 명세서

1. 요구사항 명세서(SRS - Software Requirements Specification)란

- **정의**: 소프트웨어 시스템의 기능적, 비기능적 요구사항 및 제약조건들을 체계적으로 정리한 문서이다.
- **목적**: 프로젝트 이해관계자들 간의 명확한 소통을 위해 요구사항을 문서화하고, 개발 과정의 기준을 제공한다.

2. 요구사항 명세서의 구성

- 요구사항 명세서에 내용이나 양식에 대한 표준은 없다. 업체들 마다 자신들만의 표준을 가지고 정의한다.

1. **개요 (Introduction)**:
    - **목적 (Purpose)**: SRS 문서의 목적과 범위를 기술한다.
    - **범위 (Scope)**: 시스템의 개요와 개발의 목표를 설명한다.
    - **정의 (Definitions)**: 주요 용어와 약어를 정의한다.

2. **기능적 요구사항과 비기능적 요구사항을 분리해서 정의**:

- **기능적 요구사항 (Functional Requirements)**:
  - **정의**: 시스템이 수행해야 할 기능을 구체적으로 기술한다.
  - **예**: 사용자 로그인, 판매 내역 보고서 생성 등.
- **비기능적 요구사항 (Non-Functional Requirements)**:
  - **정의**: 시스템의 품질 속성(성능, 보안, 사용성 등)을 기술한다.
  - **예**: 응답 시간, 데이터 처리 속도, 보안 규격 등.

3. **항목**

- **요구사항 ID**: 요구사항을 식별할 수있는 고유번호 또는 코드. 요구사항 추적이나 관리시 사용한다.
- **요구사항명**: 요구사항의 내용을 간략하게 표현할 수있는 이름
- **요구사항 내용**: 요구사항에 대한 자세하고 구체적인 설명.
- **중요도**: 요구사항이 프로젝트 목적에 얼마나 관계가 있는지를 나타냄. 업무 우선순위와 연결됨.
- **수용여부**: 요구사항에 대한 수용 여부 ex) 수용, 검토중, 미수용

3. 요구사항 명세서 작성 원칙

- **명확성 (Clarity)**: 요구사항을 명확하고 이해하기 쉽게 기술한다. 하나의 요구사항에는 하나의 내용만 포함시킨다.
- **완전성 (Completeness)**: 프로젝트의 모든 필요한 요구사항(기능과 조건)들을 포함한다.
- **일관성 (Consistency)**: 요구사항 간의 모순되거나 중복되는 것이 없도록 한다. 표현 방식, 용어도 일관되게 사용해야 한다.
- **검증 가능성 (Verifiability)**: 요구사항이 테스트를 통해 검증될 수 있도록 기술한다. 측정 가능하거나 관찰 가능한 기준을 제시해야 한다.
- **추적 가능성 (Traceability)**: 각 요구사항이 개발 과정에서 추적될 수 있도록 한다.

요구사항 명세서의 예

요구사항ID	기능유형	요구사항명	요구사항 내용	중요도	담당자	요청자	수용여부
REQ-001	기능	상품 검색	사용자가 키워드, 카테고리, 가격 범위로 상품을 검색할 수 있어야 함	상	김XX	이XX	수용
REQ-002	기능	장바구니 관리	사용자가 상품을 장바구니에 추가, 삭제, 수량 조절할 수 있어야 함	상	박XX	이XX	수용
REQ-003	기능	주문 처리	사용자가 장바구니의 상품을 주문하고 결제할 수 있어야 함	상	김XX	최XX	수용
REQ-004	기능	회원 관리	회원 가입, 로그인, 정보 수정, 탈퇴 기능을 제공해야 함	중	박XX	이XX	수용
REQ-005	기능	상품 리뷰	구매한 상품에 대해 별점과 리뷰를 작성할 수 있어야 함	중	김XX	최XX	수용
REQ-006	비기능	반응형 디자인	모바일, 태블릿, PC 등 다양한 기기에서 최적화된 화면을 제공해야 함	상	박XX	이XX	수용
REQ-007	비기능	보안	HTTPS를 사용하고 개인정보를 암호화하여 저장해야 함	상	김XX	정XX	수용
REQ-008	비기능	성능	페이지 로딩 시간이 3초 이내여야 함	중	김XX	이XX	검토중
REQ-009	기능	관리자 페이지	상품, 주문, 회원 관리를 위한 관리자 페이지를 제공해야 함	상	김XX	박XX	수용
REQ-010	기능	할인 쿠폰	사용자에게 할인 쿠폰을 발급하고 적용할 수 있어야 함	하	박XX	최XX	검토중
REQ-011	비기능	성능	웹사이트는 초당 1000명의 동시 접속자를 처리할 수 있어야 함	상	김XX	유XX	수용

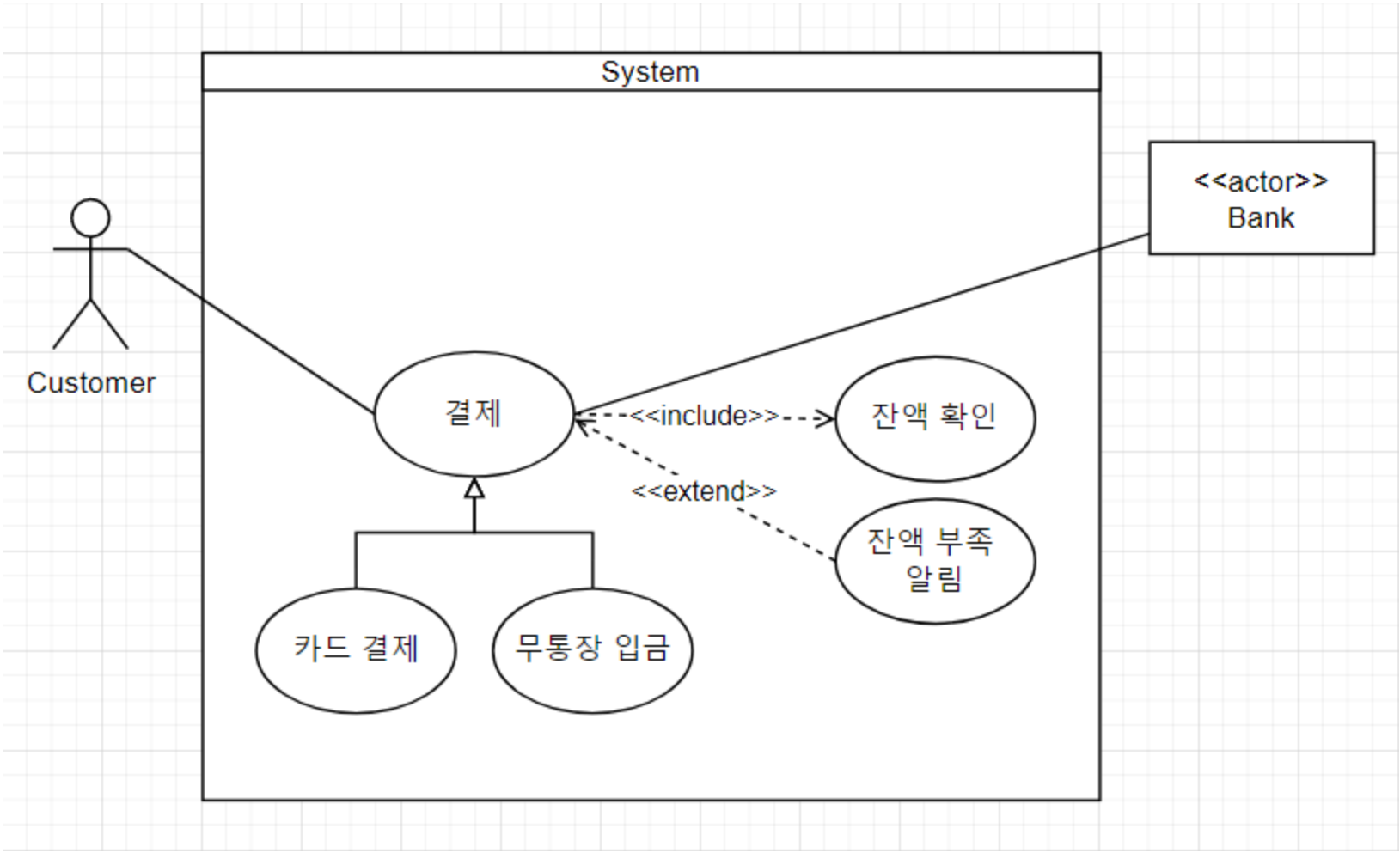
유즈케이스(User case)

유즈케이스(User case) 란

- 유스케이스란 **행위자(actor)가 관심을 가지고 있는 일을 달성하기 위해 시스템이 제공하는 시나리오의 집합을 말한다.**
- 조금 더 쉽게 표현하자면 시스템이 사용자와 상호작용하는 방식을 구체적으로 설명하는 시나리오를 의미한다. 이는 시스템의 기능 요구사항을 사용자가 이해하기 쉽게 표현하며, 소프트웨어 개발 과정에서 사용된다.

유즈케이스 산출물

- **유즈케이스 다이어그램**
  - **시스템과 사용자간의 상호작용을 다이어그램으로 표현 한 것**으로 사용자 관점에서 시스템의 기능/서비스와 그와 관련된 외부 요소와의 관계를 보여준다.



- **유즈케이스 명세서(정의서)**
  - 유즈케이스 별로 액터로 부터 시작하여 기능이 실행되는 일련의 흐름을 문서로 작성한 것.

유즈케이스의 주요 요소

1. **유즈케이스 이름 (Use Case Name)**
    - 유즈케이스의 내용을 대표하는 짧고 명확한 이름이다.
2. **행위자 (Actor)**
    - 시스템과 상호작용하는 주체를 의미하며, 사용자, 다른 시스템, 장치 등이 될 수 있다.
3. **목적 (Goal)**
    - 유즈케이스가 달성하고자 하는 목표나 목적을 설명한다.

4. **사전 조건 (Preconditions)**
    - 유즈케이스가 시작되기 전에 반드시 충족되어야 하는 조건을 명시한다.
5. **후 조건 (Postconditions)**
    - 유즈케이스가 완료된 후에 시스템이 충족해야 하는 조건을 설명한다.

6. **주요 흐름 (Main Flow)**
    - 유즈케이스가 정상적으로 진행되는 기본 시나리오를 단계별로 서술한다.
7. **대체 흐름 (Alternative Flow)**
    - 주요 흐름과 다른 상황에서 진행되는 대체 시나리오를 설명한다.
8. **예외 흐름 (Exception Flow)**
    - 에러나 예외 상황에서의 흐름을 서술한다.
9. **특별 요구사항 (Special Requirements)**
    - 성능, 보안, 제약 조건 등 특별한 요구사항이 있을 경우 명시한다.

유즈케이스의 목적과 중요성

1. **요구사항 명확화**
    - 시스템의 기능 요구사항을 명확하고 구체적으로 정의함으로써 사용자와 개발자가 같은 이해를 공유할 수 있게 한다.
2. **개발 과정 지원**
    - 유즈케이스는 시스템 설계, 개발, 테스트 과정에서 참고 자료로 사용되어 일관성과 정확성을 유지하는 데 기여한다.

3. **커뮤니케이션 도구**
    - 개발자, 디자이너, 테스터, 비즈니스 분석가 등 다양한 이해관계자 간의 원활한 의사소통을 돕는다.
4. **테스트 케이스 작성**
    - 유즈케이스는 테스트 케이스를 작성하는 데 중요한 기준이 되며, 시스템이 요구사항을 충족하는지 검증하는 데 사용된다.

화면설계서(Wireframe)

1. 화면설계서(Wireframe) 이란

- 실제 화면디자인이 시작 되기전 UI/UX 디자인의 방향을 잡는 역할을 한다. 화면에 보여지는 요소들을 정하기 위해 그리는 프로토타입이다.
- 텍스트, 선, 버튼 같은 가장 기본적인 요소들을 이용해 그린다.

2. 화면설계서의 목적

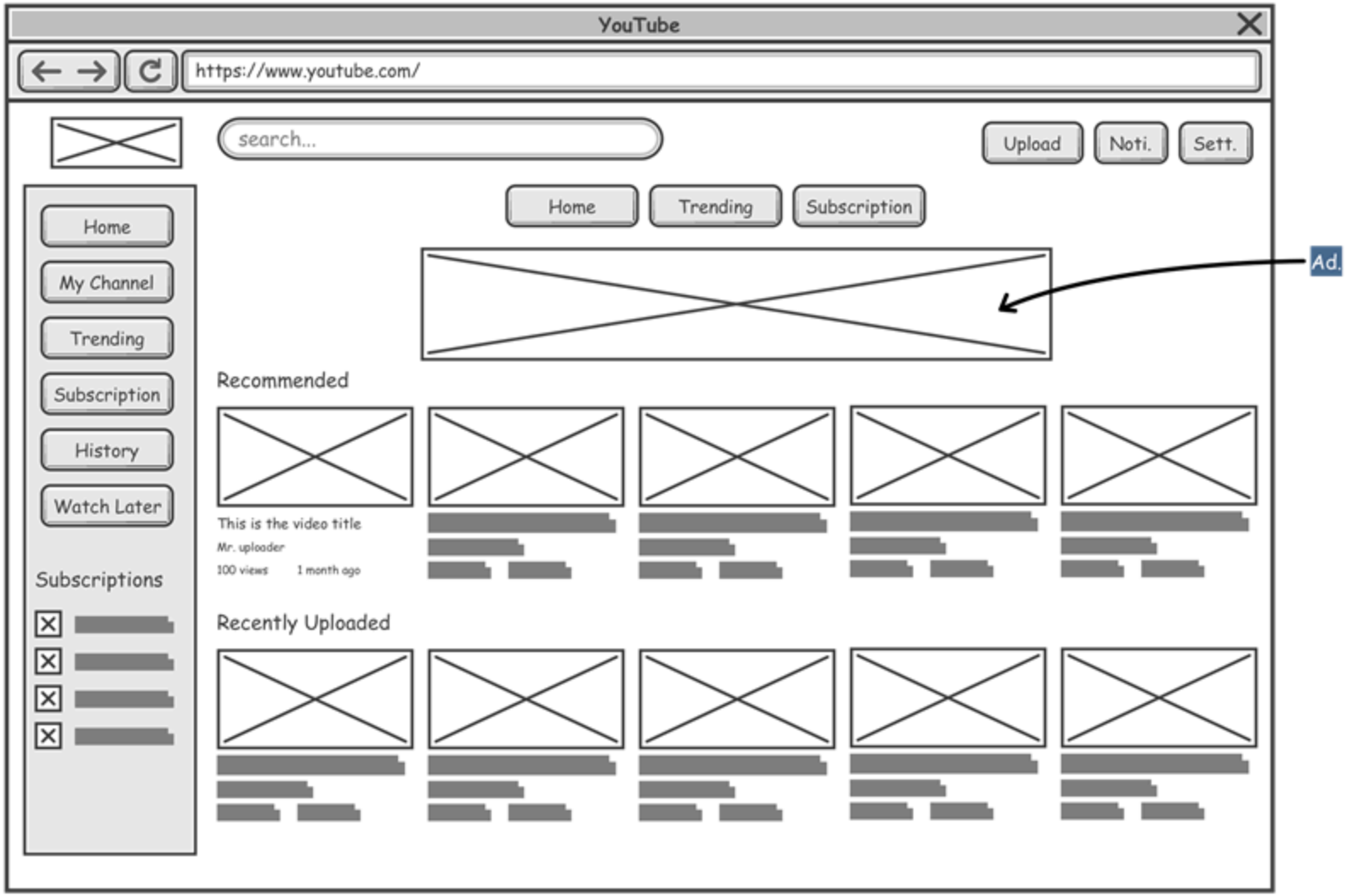
- 초기 아이디어 시각화
  - 요구사항명세서의 기능들을 시작적으로 배치하여 초기 아이디어를 시각적으로 표현한다.
- 구조와 레이아웃 결정
  - 페이지의 구조와 정보의 계층을 결정
- 팀 간 소통 도구
  - 디자이너, 개발자, 이해 관계자 간의 효과적인 소통 도구

3. 화면설계서의 중요성

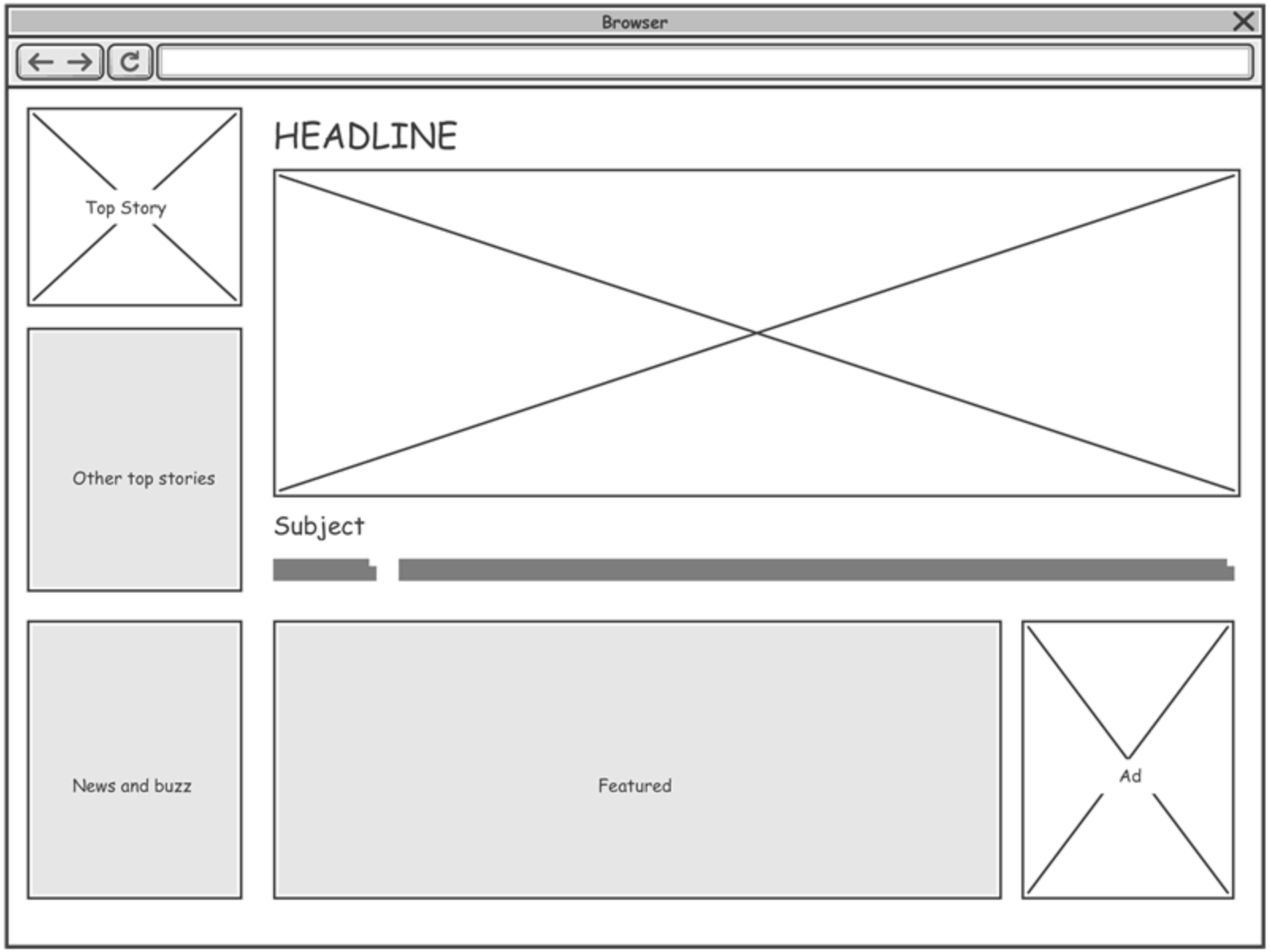
- **사용자 경험(UX) 개선:**
  - 사용자 흐름을 미리 계획하여 사용자 경험 향상
- **비용 및 시간 절약:**
  - 초기 단계에서 문제점을 발견하고 수정하여 개발 비용 절감
- **효과적인 피드백 수집:**
  - 이해 관계자 및 사용자로부터 초기 피드백을 받아 최종 디자인에 반영

4. 예

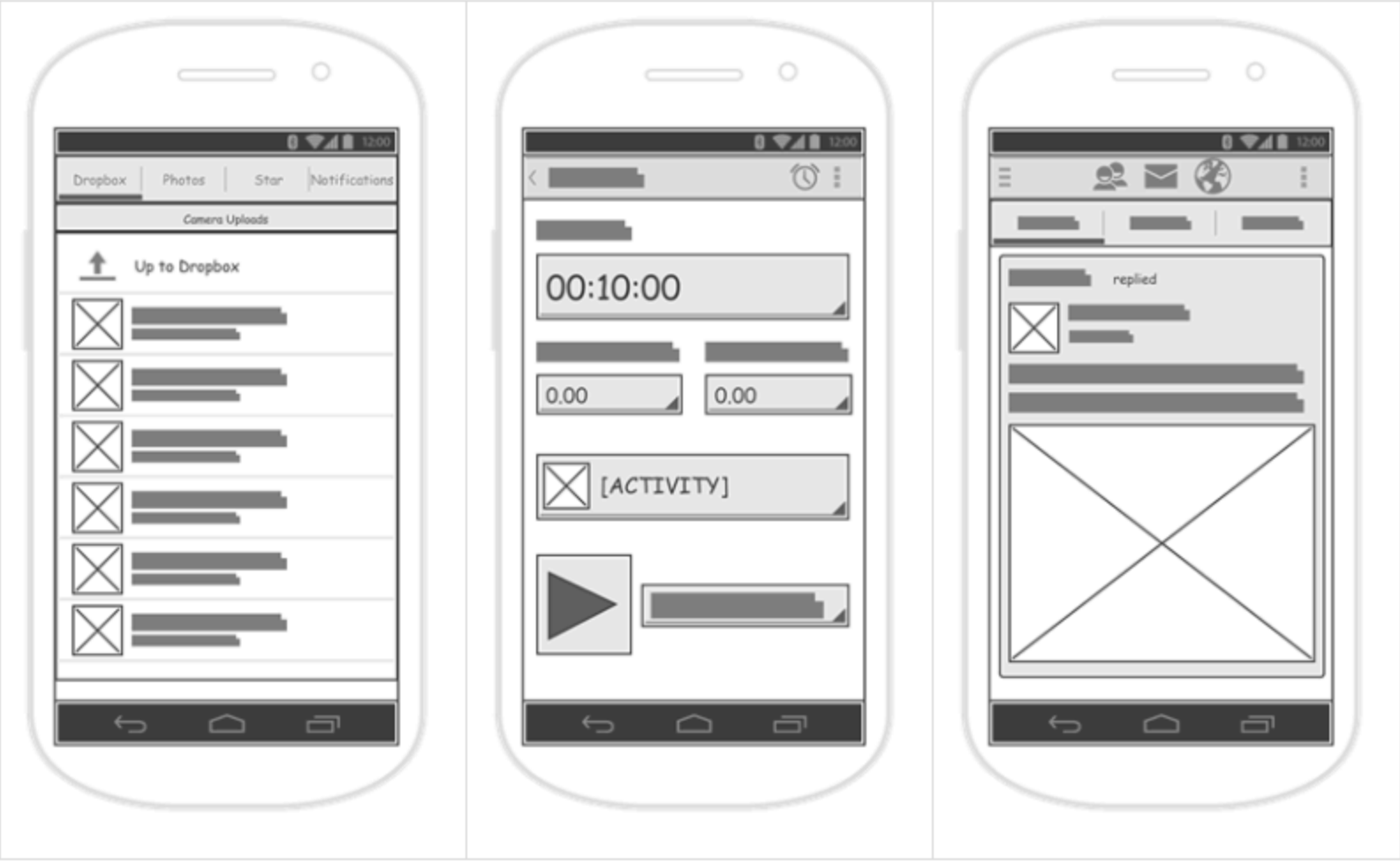
유튜브 메인페이지


YouTube
https://www.youtube.com/
search...
Upload
Noti.
Sett.
Home
Trending
Subscription
Home
My Channel
Trending
Subscription
History
Watch Later
Ad.
Recommended
This is the video title
Mr. uploader
100 views
1 month ago
Subscriptions
Recently Uploaded

신문사 홈페이지


Browser
Top Story
HEADLINE
Other top stories
Subject
News and buzz
Featured
Ad

앱 화면설계


Dropbox
Photos
Star
Notifications
Camera Uploads
Up to Dropbox
00:10:00
0.00
0.00
[ACTIVITY]
replied

화면설계서와 기능명세서

화면코드	COM-B03
화면명	Signup-User-Step2
	회원가입-사용자-2단계

프로젝트		번호	9

#	기능명
1	**사진** - 기본 사진이 첨부되어 있다.
2	**이메일 유효 검사** - 이메일 형식에 맞는지 확인 - 필수 - 소셜 회원가입자는 이메일 입력폼을 수정할 수 없다. [에러] 이미 사용된 이메일입니다.
3	**비밀번호 유효성 검사** - 비밀번호는 최소 8자리 이상, 32자리 이하로 한다. [에러] 비밀번호 8자리 이상 입력해주세요.
4	**비밀번호 재입력 검사** - 첫번째 비밀번호와 같은 문자열을 입력했는지 검사한다. [에러] 비밀번호가 맞지 않습니다. 다시 입력해주세요.
5	**이름** - 최소 1자, 최대 8자 - 화면보다 이름이 길경우 (...) 처리를 한다. [에러] 이름을 입력해주세요.
6	**보안문자** - 보안문자 라이브러리를 사용한다. - [새로고침] 버튼 선택시, 보안문자를 교체한다. [에러] 보안문자를 입력해주세요. [에러] 보안문자가 맞지 않습니다. 다시 입력해주세요.

5. 참조

- https://www.archimetric.com/what-is-wireframe/
- https://mklab-co.medium.com/작성법-화면설계서-wireframe-와-기능명세서-functional-specification-bbcff0071ea2

테스트 설계

테스트 설계 목적

- 소프트웨어의 결함을 발견하고 수정하여 품질을 보장 한다.
- 소프트웨어가 요구사항을 만족하고, 사용자에게 신뢰성을 제공하는지 확인.

테스트의 종류

- **단위 테스트 (Unit Testing)**: 개별 모듈이나 함수가 올바르게 작동하는지 확인.
- **통합 테스트 (Integration Testing)**: 두개 이상의 모듈이나 소프트웨어 구성 요소들을 결합해 함께 예상대로 작동하는지 확인하는 테스트.
- **시스템 테스트 (System Testing)**: 완전하고 통합된 소프트웨어 시스템이 전체적으로 올바르게 작동하는지 확인.
- **인수 테스트 (Acceptance Testing)**: 최종 사용자의 요구사항을 만족하는지 확인.

3. 테스트 기법

블랙박스 테스트

- 소프트웨어의 내부 구조나 작동원리를 모르는 상태에서 소프트웨어의 동작을 테스트하는 방법.
- 개발자 입장이 아닌 사용자 입장에서 소프트웨어에 대한 요구사항과 결과물이 일치하는지 테스트하는 기법.

화이트박스 테스트

- 소프트웨어 혹은 제품의 내부 구조, 동작을 세밀하게 검사하는 테스트 방식
- 개발자가 소프트웨어 또는 컴포넌트 등의 로직에 대한 테스트를 수행하기 위해 설계 단계에서 요구된 사항을 확인하는 개발자 관점의 단위테스팅 기법

4. 효과적인 테스트의 특징

- **완전성**: 모든 요구사항을 커버하도록 설계된 테스트.
- **반복 가능성**: 동일한 테스트 케이스를 반복 실행해도 동일한 결과를 얻을 수 있어야 한다.
- **독립성**: 테스트는 독립적으로 실행 가능해야 하며, 다른 테스트에 의존하지 않아야 한다.
- **재사용성**: 잘 설계된 테스트 케이스는 재사용이 가능해야 한다.

테스트 케이스 설계

- 목적: 개별 테스트 케이스를 설계하여 각 요구사항이 올바르게 구현되었는지 검증.
- 포함 항목:
  - ID: 각 테스트 케이스에 고유한 식별자 부여.
  - 테스트 케이스 이름:테스트 케이스 이름
  - 입력 데이터 (Input Data): 테스트 실행 시 사용할 입력 값.
  - 전제 조건 (Preconditions): 테스트 실행 전에 만족해야 할 조건.
  - 기대 결과 (Expected Result): 입력 데이터에 따른 예상 출력 값.
  - 실제 결과 (Actual Result): 테스트 실행 후 기대되는 상태.
  - Pass 여부: 테스트 후 성공/실패 여부.