2012

Members

Developers

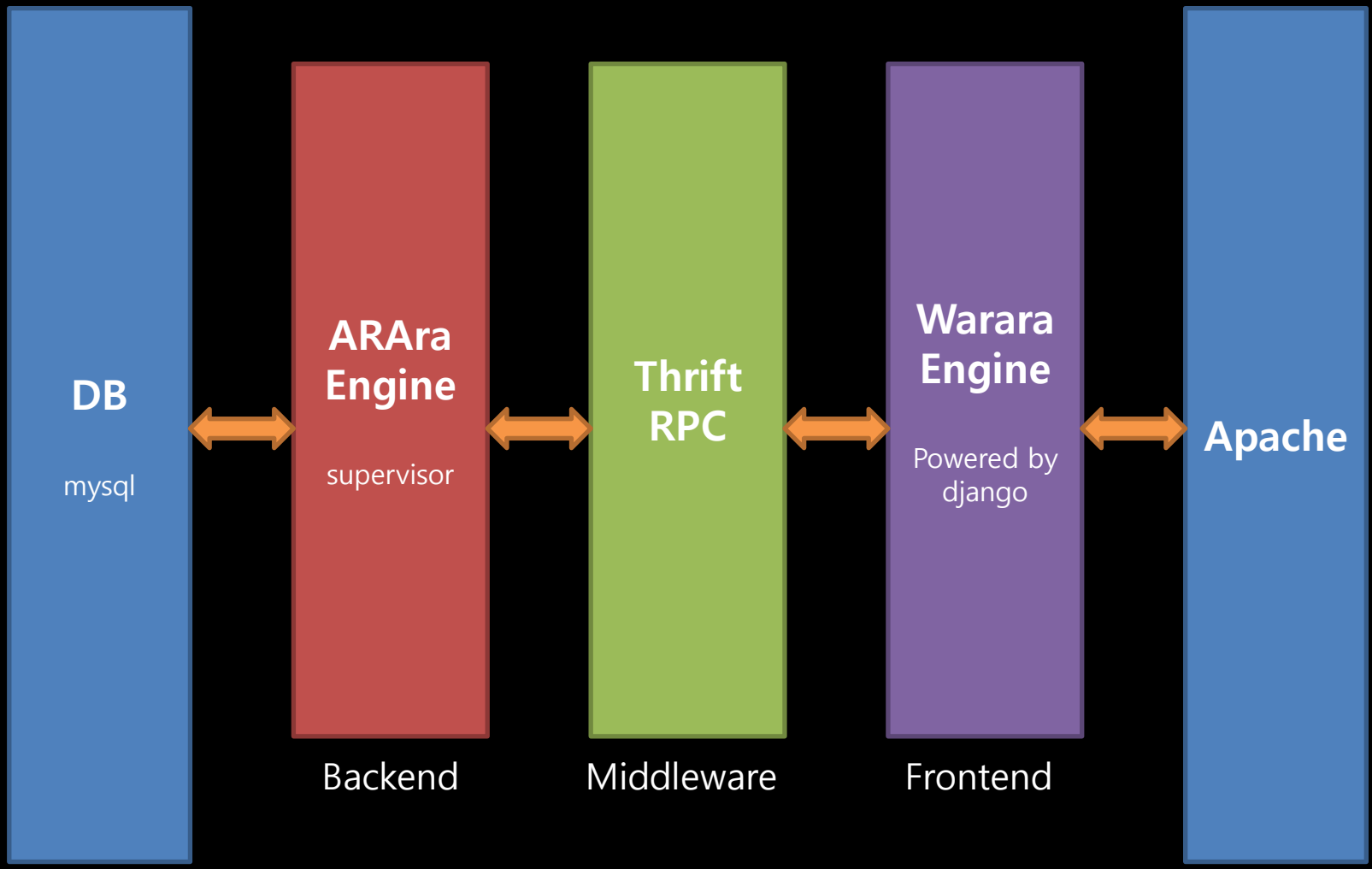
panda, richking, hodduc, bbashong,
grandmarnier, cling, penguin, pocari

Technical Advisor ..?

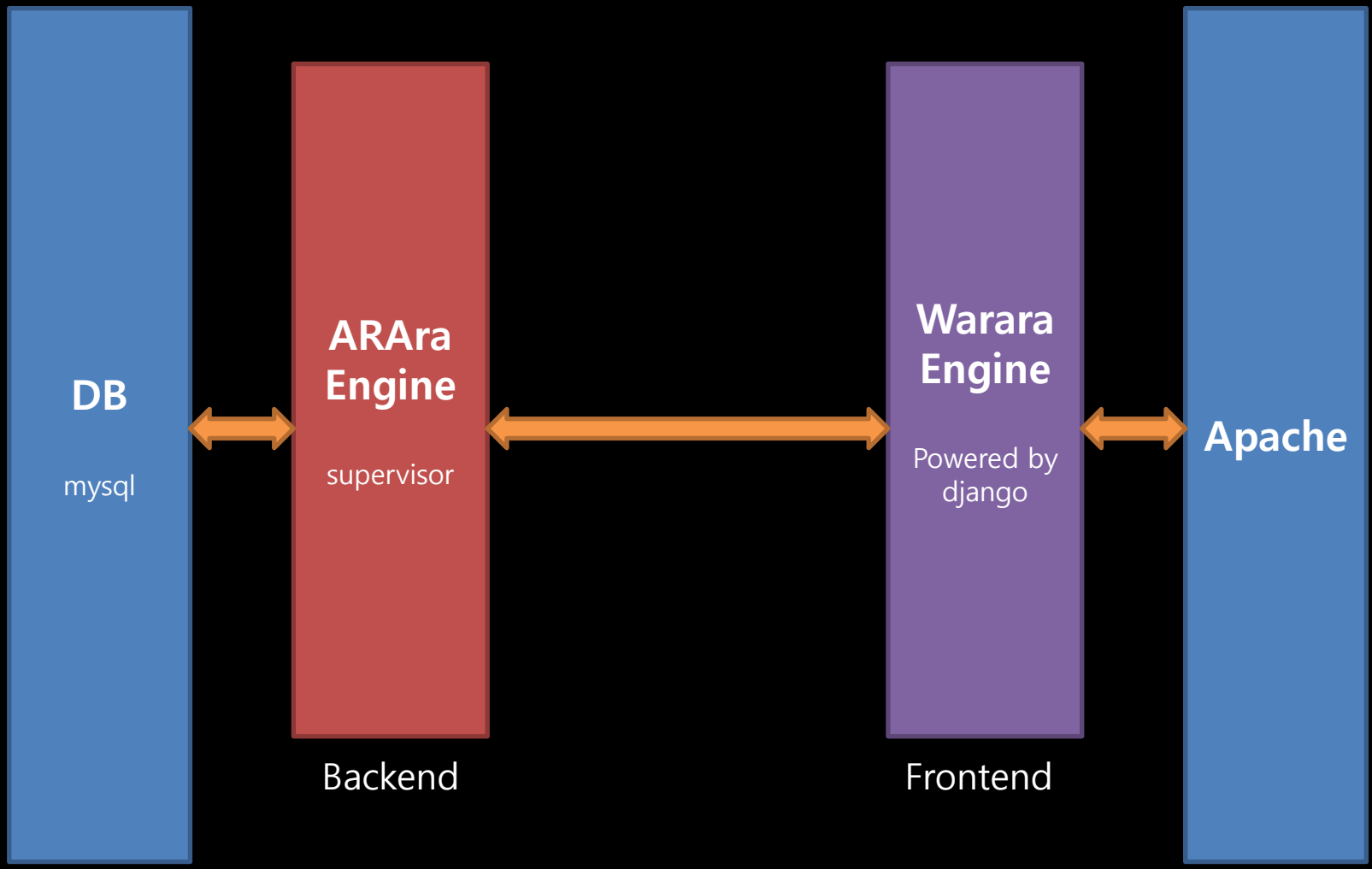
combacsa, ajmbell, acuros, zzongaly, elaborate
and more...

참가 환영!

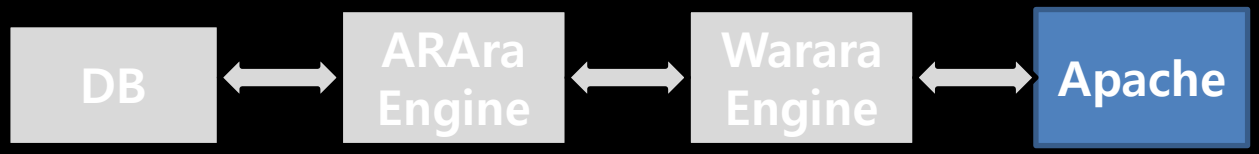
ARAra Architecture



ARAra Architecture

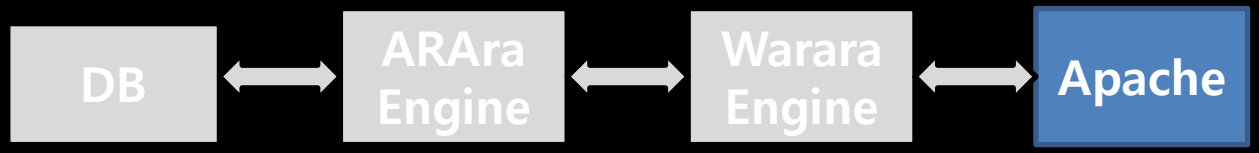


ARAra Architecture



요청:
"Garbage의 게시물 A를 추천"

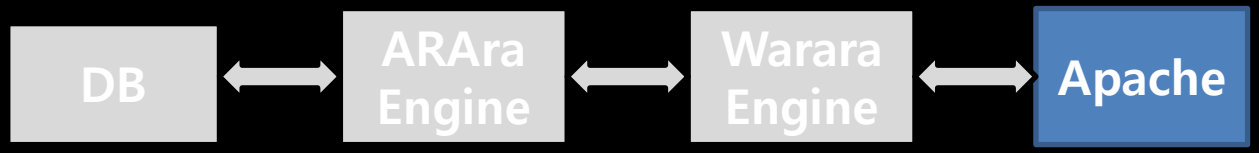
ARAra Architecture



요청:
"Garbage의 게시물 A를 추천"

http://ara.kaist.ac.kr/board/garbages/123212/123212/vote/+

ARAra Architecture

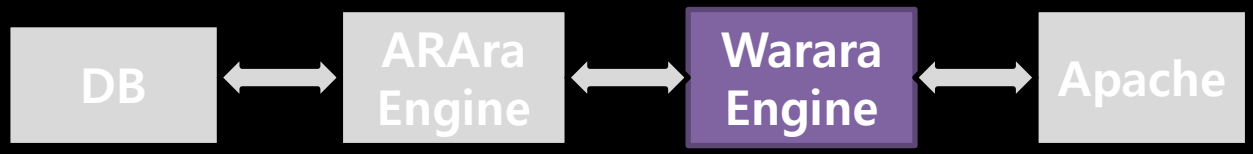


요청:
"Garbage의 게시물 A를 추천"

http://ara.kaist.ac.kr/board/garbages/123212/123212/vote/+

호출할 함수 : board.vote()
- 게시판 이름 garbages
- 스레드 번호 123212
- 글 번호 123212
- 종류 +

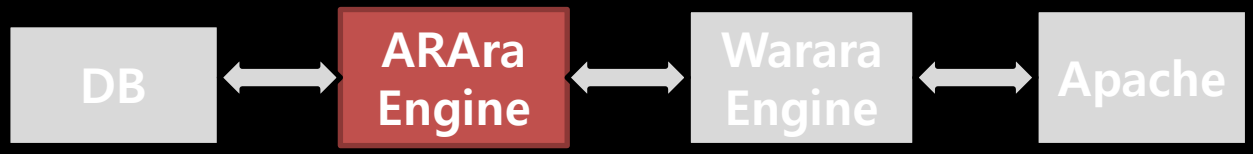
ARAra Architecture




```
def vote(request, board_name, root_id, ....):
    .......
    result = server.article_manager.vote_article(..)
    .......
    return HttpResponse("OK")
```

1. 유저가 보낸 요청(Request)을 '정리'해서
2. 적당한 형태의 Backend API를 호출한 후
3. 결과를 이쁘게 감싸서 사용자에게 전달! (Response)

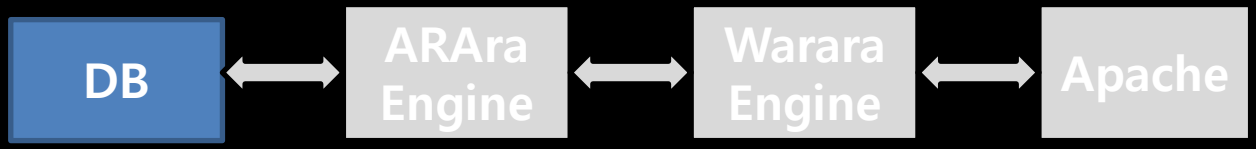
ARAra Architecture




```
Class ArticleManager(...):
  ...
  def vote_article(....):
   .......
   article.positive_vote += 1
   ......
   session.commit()
   ......
```

1. 프론트엔드가 요청한 작업을 처리하고
2. DB에 정보를 저장한 후
3. 결과를 프론트엔드로 돌려줌!

ARAra Architecture



게시판	글번호	추천수	반대수	...
	.....			
Garbages	123212	1	0	....
	.....			

model.Article

ID	username	signature	...
	.....		
Hodduc	준성	훗	...
	.....		

model.User

....

ARAra Sources

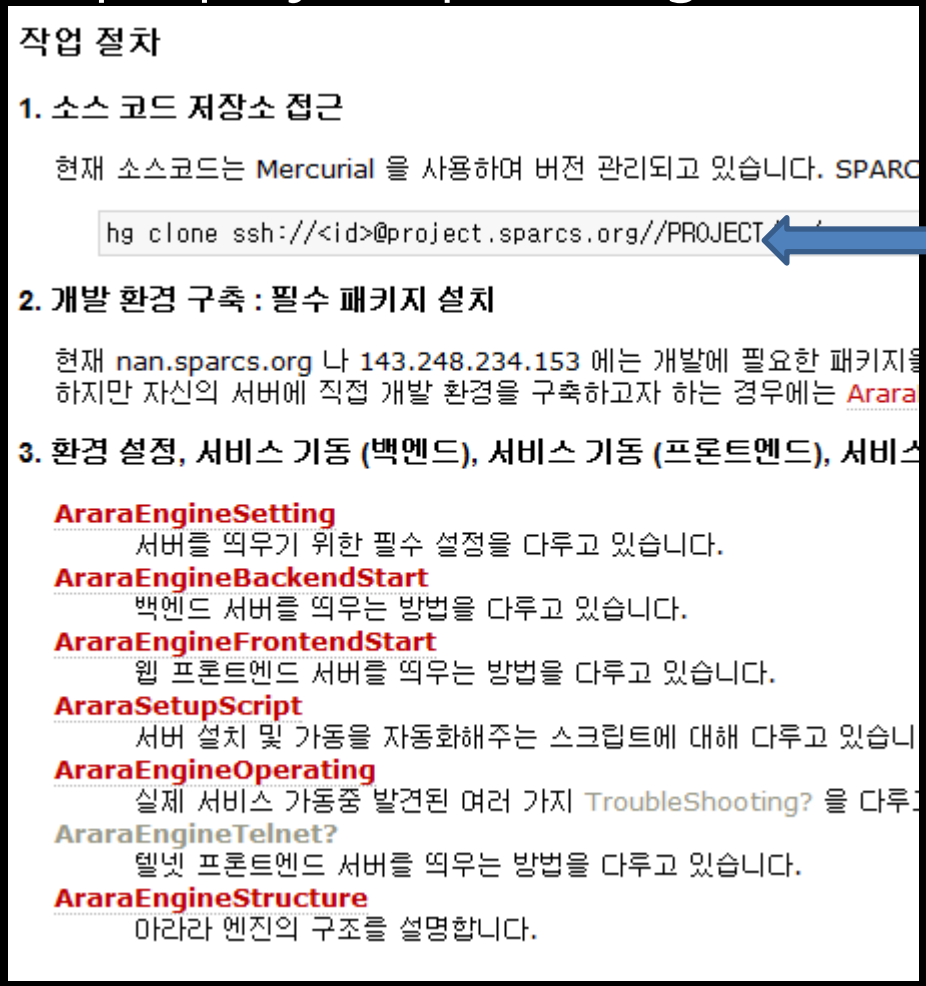
arara	아라라 백엔드 엔진. 다수의 매니저와 그에 대응하는 테스트 코드가 존재
bin	주로 아라라 엔진을 띄우는 스크립트들이 위치
etc	Arara, warara의 Config 파일이 위치
libs	Arara, warara에서 공통으로 사용하는 라이브러리 함수들이 위치
middleware	Thrift middleware의 구동 파일
tarara	Telnet Ara. 현재 사용되고 있지 않음
thirdparty	잡다한 것들
thrift	Thrift 규격 명세 파일이 들어있음
tools	잡다한 것들(2)
warara	Warara Django Project

ARAra Development Environment

Requirement	Python 2.x + Django 1.x, Thrift, MySQL, SQLAlchemy … (https://project.sparcs.org/arara/wiki/AraraEngineDependency)
VCS, ITS	Mercurial (HG) with TRAC
Main repository	ssh://project.sparcs.org//PROJECT/hg/arara
개발 서버	143.248.234.205 (ICUBE Instance)
그 외 특이사항	Reviewboard 이용 (http://review.sparcs.org/)

And More..

http://project.sparcs.org/arara/



작업 절차

1. 소스 코드 저장소 접근

현재 소스코드는 Mercurial 을 사용하여 버전 관리되고 있습니다. SPARC

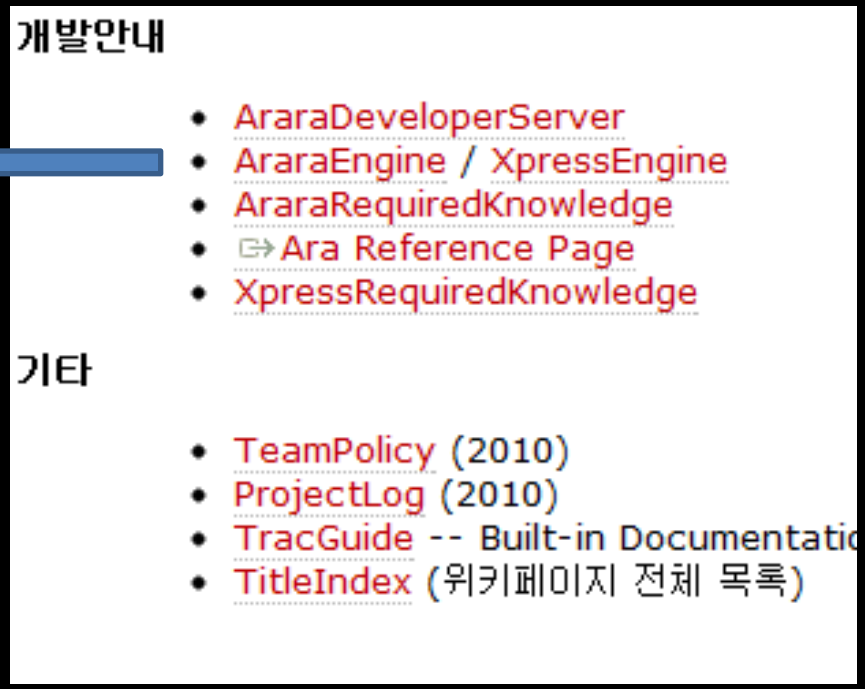
```
hg clone ssh://<id>@project.sparcs.org//PROJECT
```

2. 개발 환경 구축 : 필수 패키지 설치

현재 nan.sparcs.org 나 143.248.234.153 에는 개발에 필요한 패키지
하지만 자신의 서버에 직접 개발 환경을 구축하고자 하는 경우에는 Arara

3. 환경 설정, 서비스 기동 (백엔드), 서비스 기동 (프론트엔드), 서비스

AraraEngineSetting
서버를 띄우기 위한 필수 설정을 다루고 있습니다.
AraraEngineBackendStart
백엔드 서버를 띄우는 방법을 다루고 있습니다.
AraraEngineFrontendStart
웹 프론트엔드 서버를 띄우는 방법을 다루고 있습니다.
AraraSetupScript
서버 설치 및 가동을 자동화해주는 스크립트에 대해 다루고 있습니
AraraEngineOperating
실제 서비스 가동중 발견된 여러 가지 TroubleShooting? 을 다루
AraraEngineTelnet?
텔넷 프론트엔드 서버를 띄우는 방법을 다루고 있습니다.
AraraEngineStructure
아라라 엔진의 구조를 설명합니다.



개발안내

- AraraDeveloperServer
- AraraEngine / XpressEngine
- AraraRequiredKnowledge
- ⇨Ara Reference Page
- XpressRequiredKnowledge

기타

- TeamPolicy (2010)
- ProjectLog (2010)
- TracGuide -- Built-in Documentatio
- TitleIndex (위키페이지 전체 목록)

일반적인 개발 Flow

1. 팀 회의를 통해 일감(Ticket)을 생성
2. 담당자가 Ticket을 Accept함
3. (필요한 경우) 테스트 코드를 작성
4. 본 코드를 작성
5. 코드가 잘 작동하고 테스트에도 성공하는 것을 확인한 후 Commit 및 Post Review
6. Code Review를 통과하면 본 Repository에 반영
7. Ticket을 생성자에게 되돌려줌
8. Ticket의 생성자가 일감을 확인한 후 Close

보통은 이렇게...

1. 할 일을 정한다
2. 이 일을 하는데 필요한
   - DB 모델
   - 백 엔드 함수 리스트
   - 프론트 엔드 함수 리스트

   를 나열해본다

3. 나열된 순서대로 구현한다
4. 테스트를 통과시킨다

개발 중 반드시 지킬 것

백엔드 함수를 고치거나 생성한 경우,

반드시 해당 테스트를 **같이 고치거나 생성**할 것

모든 Revision에서 항상 **테스트에 성공**해야 함

개발 중 반드시 지킬 것

절대 **코드를 직접 Push하지 않는다**

Commit한 상태로 **Reviewboard**에 코드를 올릴 것

Review를 통과한 코드에 한해, 담당자 한 사람이 도맡아서 Push를 진행

개발 중 반드시 지킬 것

항상 Trac을 주시하고,

Ticketing에 신경 써서 작업을 진행할 것.

모든 작업은 Ticket에서 시작해서 Ticket에서 끝남

What we do?

... 디자인 개편?

기타 논의해야 할 것들

- 팀원 일정 조사
- 일일 회의 / 주간 회의 일정 잡기
- 팀 주요 근무 시간 확정
- Pair Programming ?