# 🚀 STARTWORK.md - VideoPlanet 작업 시작 가이드

**목적**: 모든 개발 작업 시작 전 필수 체크리스트와 준비 절차를 제공합니다.  
**원칙**: Plan → Do → See 사이클의 "Plan" 단계를 체계화합니다.

## 📌 Quick Start (5분 체크리스트)

# 1. 개발 서버 확인

npm run dev # http://localhost:3000

# 2. 테스트 실행 확인

npm test

# 3. 타입 체크

npm run type-check

# 4. Git 상태 확인

git status

git pull origin master

## 📚 Step 1: 필수 문서 숙지 (우선순위 순)

### 1.1 MEMORY.md 확인 ⭐⭐⭐

# 최근 작업 내역 확인 (상위 100줄)

head -n 100 MEMORY.md

* 최근 작업 내역 파악
* 현재 진행 중인 이슈 확인
* 이전 결정 사항 검토
* 해결된 문제와 패턴 학습

### 1.2 CLAUDE.md 숙지 ⭐⭐⭐

* 프로젝트 특화 규칙 확인
* 테스트 전략 (TDD vs BDD) 이해
* UI/UX 디자인 토큰 확인
* 에러 처리 표준 숙지

### 1.3 Architecture\_fsd.md 확인 ⭐⭐

* FSD 레이어 구조 이해
  + app → processes → widgets → features → entities → shared
* 의존성 규칙 확인 (상→하만 허용)
* Public API (index.ts) 원칙 이해
* 파일 배치 기준 숙지

### 1.4 Frontend\_TDD.md 참고 ⭐

* Red → Green → Refactor 사이클 이해
* 테스트 우선 개발 루틴 확인
* 의존성 제거 전략 파악

### 1.5 DEVELOPMENT\_RULES.md 참고 ⭐

* 통합 개발 지침 확인
* Git/PR 규칙 확인
* 품질 게이트 기준 확인

## 🔍 Step 2: 현재 상태 분석 (MECE 원칙)

### 2.1 프로젝트 상태 확인

# 브랜치 확인

git branch -a

# 의존성 상태

npm ls --depth=0

# 빌드 상태

npm run build

# 테스트 커버리지

npm run test:coverage

### 2.2 작업 범위 분석

* **Structure (구조)**: 어떤 파일/폴더가 영향받는가?
* **Function (기능)**: 어떤 기능이 변경/추가되는가?
* **Process (프로세스)**: 어떤 워크플로우가 바뀌는가?

### 2.3 의존성 체크

# 순환 의존성 확인

npm run check:circular

# FSD 경계 위반 확인

npm run lint:arch

## 📝 Step 3: 작업 계획 수립

### 3.1 Definition of Ready (DoR) 체크리스트

* 요구사항이 명확한가?
* 수용 기준이 정의되었는가? (3~5개)
* 테스트 시나리오가 준비되었는가?
* API 계약(있다면)이 정의되었는가?
* 디자인/목업(있다면)이 준비되었는가?

### 3.2 작업 유형별 접근법

#### 🐛 버그 수정

1. MEMORY.md에서 유사 이슈 검색
2. 실패하는 테스트 먼저 작성
3. 최소 수정으로 해결
4. 회귀 방지 테스트 추가

#### ✨ 새 기능 개발

1. FSD 레이어 결정 (entities? features? widgets?)
2. Public API 설계
3. 테스트 먼저 작성 (TDD)
4. 구현 → 리팩토링

#### ♻️ 리팩토링

1. 현재 테스트 커버리지 확인
2. 부족한 테스트 보강
3. 작은 단위로 점진적 개선
4. 각 단계마다 테스트 실행

#### 🎨 UI 작업

1. 컴포넌트 복잡도 판단
   * 복잡한 로직 → TDD (테스트 먼저)
   * 단순 표시 → 구현 후 즉시 테스트
2. 디자인 토큰 확인
3. 접근성 체크리스트 준비

## 🛠 Step 4: 개발 환경 준비

### 4.1 필수 스크립트 확인

# 개발 서버 (포트 3000)

npm run dev

# 백엔드 상태 확인

curl https://videoplanet.up.railway.app

# Storybook (있다면)

npm run storybook

### 4.2 브라우저 개발자 도구

* React DevTools 설치
* Redux DevTools 설치
* Network 탭 열어두기
* Console 에러 확인

### 4.3 VS Code 설정

// 권장 익스텐션 확인

- ESLint

- Prettier

- TypeScript and JavaScript Language Features

- GitLens

- Error Lens

## ✅ Step 5: Pre-Work 체크리스트

### 최종 확인 (모두 체크되어야 작업 시작)

* MEMORY.md 최신 내역 확인 완료
* 관련 개발 지침 숙지 완료
* 현재 Git 상태 clean (uncommitted 변경사항 없음)
* 모든 테스트 통과 중
* 타입 체크 통과
* 개발 서버 정상 작동
* 작업 목표와 범위 명확
* 테스트 시나리오 준비 완료

## 🎯 Step 6: 작업 시작 선언

### TodoWrite 도구로 작업 계획 등록

// 예시: 3개 이상의 복잡한 작업인 경우

TodoWrite([

{ id: "1", content: "기존 코드 분석 및 테스트 확인", status: "pending" },

{ id: "2", content: "실패하는 테스트 작성", status: "pending" },

{ id: "3", content: "기능 구현", status: "pending" },

{ id: "4", content: "리팩토링 및 최적화", status: "pending" },

{ id: "5", content: "MEMORY.md 업데이트", status: "pending" }

])

### 작업 시작 커맨드

# 작업 브랜치 생성 (선택사항)

git checkout -b feat/작업명

# 또는 master에서 직접 작업

git checkout master

git pull origin master

## 📊 Step 7: 작업 중 체크포인트

### 30분마다 확인

* 테스트가 여전히 통과하는가?
* 의도한 범위를 벗어나지 않았는가?
* 커밋할 만한 단위가 완성되었는가?

### 1시간마다 확인

* FSD 레이어 경계를 지키고 있는가?
* 불필요한 의존성을 추가하지 않았는가?
* 성능 영향은 없는가?

## 🏁 Step 8: 작업 완료 절차

### 완료 전 체크리스트

* 모든 테스트 통과
* 타입 체크 통과
* 린트 경고 없음
* 커버리지 목표 달성 (전체 70%+, 핵심 85%+)
* MEMORY.md 업데이트 준비
* PR 체크리스트 준수

### MEMORY.md 업데이트 템플릿

## 📅 [날짜] [작업 제목]

### 📋 요청 사항

- [원본 요구사항]

### ✨ 구현 내용

- [주요 변경사항 1]

- [주요 변경사항 2]

### 🔍 핵심 결정사항

- [기술적 선택과 근거]

### 📁 변경된 파일

- `path/to/file1.ts` - [변경 내용]

- `path/to/file2.tsx` - [변경 내용]

### 📊 결과

- [테스트 결과]

- [성능 영향]

- [알려진 이슈]

---

## 🚨 트러블슈팅 가이드

### 자주 발생하는 문제와 해결법

#### 1. 미들웨어 오류

# middleware.ts 확인

cat middleware.ts

# 필요시 비활성화

mv middleware.ts middleware.ts.disabled

#### 2. 타입 에러

# 캐시 정리

rm -rf node\_modules/.cache

rm -rf .next

npm run build

#### 3. 테스트 실패

# 단일 테스트 실행

npm test -- --testNamePattern="테스트명"

# 디버그 모드

npm run test:debug

#### 4. Git 충돌

# 최신 변경사항 가져오기

git fetch origin

git rebase origin/master

## 📞 도움 요청

문제가 지속되면:

1. MEMORY.md에서 유사 사례 검색
2. 관련 ADR 문서 확인 (docs/architecture/)
3. 팀 슬랙 채널에 문의
4. GitHub Issues 생성

**마지막 업데이트**: 2025-08-19  
**문서 버전**: 1.0.0  
**관리자**: Claude AI Assistant

## 🎬 Quick Reference

### 필수 명령어 모음

# 개발

npm run dev # 개발 서버 시작

npm run build # 프로덕션 빌드

npm run start # 프로덕션 서버

# 테스트

npm test # 테스트 실행

npm run test:coverage # 커버리지 확인

npm run test:watch # 감시 모드

# 코드 품질

npm run lint # ESLint 실행

npm run lint:arch # FSD 경계 검증

npm run type-check # TypeScript 검증

npm run check:circular # 순환 의존성 체크

# Git

git status # 상태 확인

git add . # 스테이징

git commit -m "메시지" # 커밋

git push origin master # 푸시

### 파일 구조 빠른 참조

src/

├── app/ # 라우팅 (Next.js)

├── processes/ # 복합 플로우

├── widgets/ # 재사용 블록

├── features/ # 사용자 액션

├── entities/ # 도메인 모델

└── shared/ # 공통 유틸

**"좋은 시작이 반이다" - 체계적인 준비로 안정적인 개발을!** 🚀