



개발자, 채용 가이드북



성공적인 소프트웨어
개발자 채용을 위한
Tech HR 매뉴얼



이노베이션 아카데미



개발자, 채용 가이드북

공공누리 이용약관 안내

공공누리의 제 1유형 안내

① 이용자가 가지는 권리

1. 온·오프라인 상에 공유 및 이용 : 온·오프라인을 통하여 공유 및 이용 가능
2. 저작물 변경 : 2차적 저작물 로 변경하여 이용 가능
3. 이 저작물은 영리 목적으로 이용할 수 있습니다.

② 저작물 사용 조건

출처 표시 : 저작물의 출처를 표시하여야 합니다.

※ 공공기관이 후원한다거나 공공기관과 특수한 관계에 있는 것처럼 제 3자가 오인하게 하는 표시를 해서는 안됩니다.

③ 알아야 할 사항

1. 이용조건의 표시 및 변경

- 이용자가 공공누리 저작물 활용시 출처표시를 꼭 해주셔야 합니다.
- 공공누리 저작물의 이용조건은 변경될 수 있습니다. 다만 이용자가 이용조건 변경전 사용하셨다면 해당저작물 한해 용도변경 없이 계속 이용할 수 있습니다.

2. 이용조건의 위반

- 이용자가 공공누리 이용조건을 위반할 경우 그 즉시 이용허락이 종료됩니다.
- 이용자가 이용조건 위반 후 지속적으로 공공저작물을 이용할 경우 저작권 침해가 성립되므로 형사상, 민사상 책임을 부담 하실 수 있습니다.

3. 공공기관의 면책

- 공공기관은 공공저작물의 정확성이나 지속적인 제공 등을 보장하지 않습니다.
- 공공기관 및 그 직원은 이용자가 공공저작물을 이용함으로써 발생할 수 있는 어떠한 손해나 불이익에 대해서도 책임을 지지 않습니다.

4. 기타

- 이용자가 저작물을 보유하고 있는 공공기관으로부터 별도의 이용허락을 받은 경우 공공누리 조건을 적용하지 않으셔도 됩니다.

공공누리 제 1유형의 이용조건

공공누리에 따라 이용자는 다음에서 제시하는 조건을 준수할 경우 공공저작물을 상업적 활용 여부에 관계없이 무료로 자유롭게 이용하고 2차적 저작물 작성 등 변형하여 이용할 수 있습니다.

1 출처 표시 의무

1. 이용자는 공공저작물을 이용할 경우, 다음과 같이 출처 또는 저작권자를 표시해야 합니다.

예 “본 저작물은 ‘000(기관명)’에서 ‘00년’ 작성하여 공공누리 제0유형으로 개방한 ‘저작물명(작성자:000)’을 이용하였으며, 해당 저작물은 ‘000(기관명), 000(홈페이지 주소)’에서 무료로 다운받으실 수 있습니다.”

※ 위 내용은 예시이므로 작성연도 및 해당 기관명과 홈페이지 주소, 작성자명 기입

2. 온라인에서 출처 웹사이트에 대한 하이퍼링크를 제공하는 것이 가능한 경우에는 링크를 제공하여야 합니다.
3. 이용자는 공공기관이 이용자를 후원하거나 공공기관과 이용자가 특수한 관계에 있는 것처럼 제3자가 오인하게 하는 표시를 해서는 안됩니다.

2 이용이 제한되는 정보

다음과 같은 정보는 공공저작물에 포함되어 있더라도 관련 법령에 따라 이용이 제한됩니다.

1. 개인정보보호법, 정보통신망 이용촉진 및 정보보호에 관한 법률 등에서 보호하는 개인정보
2. 신용정보의 이용 및 보호에 관한 법률 등에서 보호하는 신용정보
3. 군사기밀보호법 등에서 보호하는 군사기밀
4. 상표권, 디자인권, 특허권 등 다른 권리의 대상이 되는 정보 또는 제3자의 저작권 등이 있는 정보
5. 기타 다른 법령에 따라 이용이 제한되는 정보

3 공공기관의 면책

1. 공공기관은 공공저작물의 정확성이나 지속적인 제공 등을 보장하지 않습니다.
2. 공공기관은 이용자가 공공저작물을 이용함으로써 발생할 수 있는 어떠한 손해나 불이익에 대해서도 공공기관 및 그 직원은 책임을 지지 않습니다.

4 이용허락조건 위반의 효과

이용자는 공공누리의 이용허락의 조건 중 어느 하나라도 위반한 경우 이용허락이 자동으로 종료되며, 이용자는 즉시 공공저작물의 이용을 중단해야 합니다.



학장님 인사말

서기 2000년 하고도 20년이 더 지났습니다. 언젠가부터 검색이 있더니 예상대로 모든 회사가 ○○을 만들거나 ○○ 서비스를 제공하는 소프트웨어 회사가 되었습니다. 회사가 성장하기 위한 동력의 상당 부분을 소프트웨어 개발자가 담당한다는 뜻입니다. 이제 그 규모나 업종에 상관없이 개발자와 친한 회사, 개발자를 잘 뽑는 회사, 개발자를 중히 여기는 회사, 개발자가 성장할 수 있는 회사가 되는 법을 배워야 합니다.

소프트웨어 개발 업무 및 소프트웨어 개발자는 다른 업무 및 다른 인력과는 다릅니다. 그리고 모든 분들이 100% 공감하는 바와 같이 절대적으로 소프트웨어 개발자가 부족합니다. 채용을 잘 하고 그 인력을 잘 운영하기 위한 세심한 준비와 실행이 필요합니다.

이노베이션 아카데미는 소프트웨어가 회사의 가치를 결정한다고 믿고 소프트웨어 개발자를 채용하여 그들과 함께 성장하고 싶은 회사들과 그 구체적인 방법을 공유하기 위하여 이 매뉴얼을 만들었습니다.

이 매뉴얼로 모든 회사가 소프트웨어로 세상을 바꾸려는 용기와 큰 힘을 얻기를 기대합니다.

2020년 12월

이노베이션 아카데미 학장 이 민석

지난 10년간 사회는 우리가 예상치 못한 빠른 속도로 발전했다. 2009년 애플 아이폰의 등장으로 인해 ‘모바일 혁명’ 시대가 시작되었고, 소셜 미디어라는 새로운 형태의 서비스가 등장하면서 변화의 속도는 더욱 빨라졌다. 이러한 변화로 인해 스마트폰으로 대체할 수 있는 대부분의 산업은 내리막길을 걸었다. 이후 4차 산업혁명 시대라는 용어와 함께 인공지능, 빅데이터, 사물인터넷, 클라우드 등이 주요 기술로 주목받으면서 사회는 또 한 번 급변하기 시작했다. 국내외 시가총액 상위 기업은 소프트웨어/IT 중심의 기업들이 자리를 차지하고 있으며 제조업 등의 전통적인 기업은 IT 기반으로 디지털 전환을 추진하고 있다. 그만큼 소프트웨어와 IT의 중요성은 이제 아무리 강조해도 지나치지 않는 시대가 되었다.

이 기술을 움직이는 핵심 인력은 누구일까? 바로 ‘개발자’라는 직종이다. 여러분은 개발자라는 말을 처음 들어봤을 수도 있다. 2019년 기준 성인의 스마트폰 하루 평균 사용 시간은 3시간 55분이라고 한다. 스마트폰을 그냥 들고 있는 시간이 아니다. 웹 서핑, 메신저, 이메일, 게임 등 다양한 일을 스마트폰으로 한다. 그럼 여러분이 지금도 활용하고 있는 수많은 앱은 누가 만든 것일까? 바로 개발자들이다. 내일부터 스마트폰을 못 쓰고, 예전에 쓰던 2G폰을 사용한다고 가정해보자. 여러분의 생활은 얼마나 불편해질까? 상상만 해도 끔찍하다. 이렇게 사회 대부분의 서

비스를 만드는데 개발자는 반드시 필요한 존재가 되었다, 앞으로도 이러한 수요는 더욱 늘어날 것이다. 하지만 회사에서 제대로 된 개발자를 뽑는 것은 ‘하늘의 별 따기’ 수준이다. 이러한 문제는 비단 스타트업/중소기업만의 문제가 아니다. 대기업도 개발자 채용을 위해 사활을 걸고 있으나 그만큼 역량을 가진 개발자를 채용하지 못하고 있다. 말 그대로 ‘개발자 채용 전쟁’ 시대인 것이다. 하지만 ‘우리 회사는 개발자가 필요 없어’라고 생각할지도 모른다. 하지만 이러한 생각은 기업 경쟁력 약화로 이어지고 결과적으로 기업이 문을 닫을 수도 있다는 생각을 해야 한다.

이렇게 개발자가 중요하고, 이들을 채용하기가 매우 어려운 시점에서 어떻게 하면 유능한 개발자를 잘 채용할 수 있을까? ‘우리 기업은 규모가 작아서 채용하기 어렵지 않을까’, ‘우리는 대기업에 비해 연봉이나 복지가 부족한데 좋은 개발자 채용이 어려울 것 같아’라고 생각할 수 있다. 본 가이드북은 이렇게 개발자 채용에 어려움을 느낄 수 있는 스타트업, 중소기업의 대표/경영진/인사담당자들을 위해서 제작했다. 현장의 목소리를 더욱 잘 반영하기 위해 관련 분야의 경험이 풍부한 여러 기관과 전문가가 참여하였고, 본 가이드북의 완성도를 높이는 데 큰 도움을 주셨다. 우선 국내 최고의 소프트웨어 혁신 인재 양성 기관인 ‘이노베이션 아카데미’서 제작 지원 및 전문가 의견 수렴에 도움을 주셨으며, 구인 회사와 구직자의 입장을 가장 잘 이해하고 있는 Tech HR 전문 기업 ‘eBrain’에서 개발자 채용에 관한 현실적인 조언과 최근 채용 트렌드, 채용 관련 양식 등을 지원해주셨다. 또한, 국내 최대 오픈소스 개발자 커뮤니티인 ‘OKKY’에서 실제 구직자 입장의 다양하고 소중한 의

견을 전달해주셨다. 이렇게 값진 의견이 반영된 소중한 가이드북인 만큼 여러분은 이 가이드북을 잘 활용하여 더욱 유능한 개발자를 채용하고 이를 기업의 성장으로 연결하는 기회를 마련하기 바란다.

이 책은 크게 5개의 장으로 구성되었다. 1장에서는 IT 기술 기반의 사회 변화, 개발자 채용의 어려움 등 사회적 배경에 대해 다룬다. 2장에서는 개발자를 맞이하기 위해 필요한 다양한 준비 과정을 소개한다. 구체적으로 채용 마케팅, 소프트웨어 산업, 개발자들을 이해할 수 있는 기초적 내용에 대해서 소개한다. 3장에서는 개발자들을 뽑는 방법에 대해 소개한다. 목적에 맞는 채용을 준비하는 과정과 채용을 실제로 실행하는 4단계에 대해서 구체적으로 소개한다. 4장에서는 이렇게 채용한 개발자를 잘 케어할 수 있는 방법에 대해 소개한다. 개발자들이 오래 머물고 싶은 회사를 만드는 방법과 지속적으로 소통하는 방법, 팀 관리법 등에 대해서 구체적으로 다룬다. 5장(부록)에서는 채용자가 알아야 할 팁과 사례, 기업에서 개발자 채용 시 활용할 수 있는 다양한 형식들을 다룬다. 가이드북의 분량이 많다고 생각하면 필요한 부분만 골라서 읽어 도 좋다. 또한, 각 장의 마지막에는 요약 형식의 액션 아이템을 삽입했다. 정 시간이 없다면 그 부분만을 위주로 읽어 나가는 것을 추천한다. 그리고 책의 모든 내용은 누구나 Github을 통해서 확인할 수 있다. 아무쪼록 이 가이드북을 통해 여러분의 회사가 빠른 디지털 전환과 4차 산업혁명 시대의 핵심 기업으로 발돋움할 수 있는 계기가 되길 바란다.



<https://github.com/innovationacademy-kr/tech-hr>

차례

학장님 인사말	4
프롤로그	5

PART 1

개발자 채용 시장에 대한 이해

01 10년 전과는 다른 IT 기업	12
변화의 핵심 동력은 IT 기술	12
시가총액 TOP 10의 변화	15
IT 기업으로 변신은 무죄	19
IT 기술의 핵심은 개발자	21
02 개발자 채용 전쟁	23
개발자 채용을 위한 소리 없는 전쟁	23
개발자에 대한 인식 변화	28
주요 IT 기업 개발자 채용 현황	34
스타트업의 개발자 채용 비율	34
03 개발자가 ‘갑’ 이다	37
잠시 머물다 가는 개발자들	37
항상 채용하는 마음가짐을 가져라	42
2020년 채용 트렌드	45

PART 2

개발자를 맞이할 준비는 되어있는가?

01 좋은 인재 채용을 위한 마케팅	52
채용 마케팅은 무엇일까?	52
채용 브랜딩을 통해 유능한 인재를 사로잡자	56
소프트웨어 산업의 이해	66
02 Selling Point를 잡아라	75
개발자, 당신은 누구인가?	75
개발자들에게 알려라	86
Developer Relations(이하 DevRel)	90

PART 3

어떻게 뽑을 것인가?

01 채용 준비하기	96
채용 목적을 명확히 하라	97
목적에 맞는 개발자 구분/채용 형태 정하기	102
채용담당자 정하기	111
채용 채널의 다양화	116
02 채용 실행하기	118
Sourcing	118
Screening	127
Interview	133
Offer	148

PART 4

관리보다는 케어하라

01 떠나기 싫은 회사 만들기	156
개발자에게 집중하는 것은 당신 몫이다	157
동기는 개발자를 춤추게 한다	162
개발자도 멘토링이 필요하다	164
02 개발자들과 소통하는 법	168
소통의 문화 이해하기	168
신뢰 관계 만들기	176
03 팀의 성장은 회사의 성장이다	182
개발팀 운영하기	182
함께 일하는 방법	193

APPENDIX

부록

01 고용자가 알아야 할 팁과 사례들	204
대표들이 전하는 대표들을 위한 팁	204
국내외 채용 팁들과 사례	206
디지털 전환 성공 사례	208
02 기업에서 사용할 수 있는 형식들	211
채용에 관한 여러 가지 형식들	211
DevRel과 관련된 프로세스	212
에필로그	215



1.

개발자 채용 시장에 대한 이해

01 | 10년 전과는 다른 IT 기업

02 | 개발자 채용 전쟁

03 | 개발자가 '값' 이다



01

10년 전과는 다른 IT 기업

바야흐로 4차 산업혁명 시대가 도래하였다. 최근 10년간 사회는 우리가 생각하지 못했던 아주 빠른 속도로 발전해왔다. 그 이면에는 다양한 정보통신기술(이하 IT 기술)의 발전이 있었다. 이러한 기술 덕분에 우리는 이전보다 훨씬 풍족하고 윤택한 삶을 영위하고 있다.

변화의 핵심 동력은 IT 기술

그렇다면 과거 30년 동안 무슨 일이 있었던 것일까? 우리 생활에서 접할 수 있는 주요 기술의 발전을 중심으로 과거의 변화를 살펴보자.

2009년 애플 아이폰의 등장으로 인해 일명 ‘모바일 혁명’ 시대가 시작되었다. 이로 인해 스마트폰이 대중화되었으며 우리 생활을 편리하게 바꾸어주는 다양한 애플리케이션도 봇물 터지듯 쏟아졌다. 여기에 소셜 미디어라는 새로운 형태의 서비스가 등장함으로써 이러한 변

화 속도는 더욱 빨라졌다.

‘트위터’, ‘페이스북’으로 촉발된 소셜 미디어 서비스는 전 세계인의 소통 방식을 바꾸었고, 정보 생산의 주체가 개인이 되는 시대가 되었다. 지금도 사람들은 다양한 소셜 미디어를 통해 생활, 뉴스, 다양한 정보 등을 공유하고 있다. 이러한 변화로 인해 모바일(스마트폰) 분야의 기업은 급성장하였고 스마트폰으로 대체 가능한 산업은 내리막길을 걸었다. 대표적으로 아이리버(MP3), 아이나비(내비게이션), 아이스테이션(PMP) 등의 기업은 대부분 매출이 급감하여 사업을 전향하거나 파산 선고를 받았다.

또한, 모든 사물이 연결되는 초연결 시대에도 빠르게 근접하고 있다. 초연결 시대의 핵심은 사물 인터넷(IoT) 기술이며 이 기술은 이제 우리 가정에서도 손쉽게 활용할 수 있는 기술이 되었다. 우리 가정에서 사용하는 냉장고, 세탁기, 공기청정기 등의 가전과 냉/난방 기기, 스마트폰, 그리고 AI 스피커가 연결되어 모든 제품들을 어디에서나 손쉽게 제어할 수 있는 시대가 왔다.

이러한 시대적 흐름에 따라 모바일 기기, IoT 기기 등 다양한 곳에서 엄청난 양의 데이터가 생성되었다. 이렇게 방대한 양의 데이터를 ‘빅데이터’라고 부르며, 데이터가 많아지면서 이를 수집하고 처리해야 할 필요성은 점점 커졌다. 따라서 이와 관련된 산업이 각광을 받기 시작했다. 예를 들어, 카드사에서는 고객 결제 데이터를 활용하여 새로운 수익 모델을 창출하고 콘텐츠 제공 업체는 고객의 데이터를 기반으로 맞춤형 콘텐츠와 광고 등을 제공하는 식이다.

2016년 3월에는 '세기의 바둑 대결'로 불리는 이세돌 9단과 알파고의 바둑 대결이 벌어졌다. 그 대결은 인공지능의 완승으로 끝났다. 이 사건 이후로 인공지능에 대한 일반인들의 관심이 급증했고 인공지능 관련 알고리즘과 관련 응용 제품들이 쏟아졌다. 이로 인해 최근 출시되는 가전제품 중 상당수의 제품에 인공지능 관련 기능이 탑재됐다.

'공간 학습을 통해 적절한 냉방 온도를 조절하는 인공지능 에어컨', '세탁물을 자동으로 분석하여 알맞은 코스를 선택하는 인공지능 세탁기', '보관하고 있는 식자재를 자동으로 인식하여 관련 레시피를 제공하는 냉장고' 등 우리 생활에서 활용하는 가전제품에 인공지능 기술이 널리 활용되고 있다.

이제는 우리가 원하는 관광지를 집에서도 가볼 수 있는 시대가 왔다. 바로 확장현실기술(XR, Extended Reality) 덕분인데, 이 기술은 기존의 가상현실(VR), 증강현실(AR), 혼합현실(MR)이라고 부르던 기술을 통칭하는 용어다. 해당 기술을 통해 가기 어려운 관광지에도 가볼 수 있고 현실에서 만나지 못하는 사람을 만날 수도 있다. 또한 '포켓몬고'와 같은 게임을 통해 현실감 넘치는 게임을 할 수도 있다. XR 기술을 통해 우리는 이전에 체험하지 못했던 새로운 경험을 손쉽게 맛볼 수 있는 환경이 마련되었다.

그렇다면 이러한 사회의 변화를 가져온 핵심 기술은 무엇일까? 3차 산업혁명을 이끌었던 요인은 컴퓨터 등의 IT 기술이었다. 4차 산업혁명도 이러한 IT 기술의 발전으로 이루어졌다고 해도 과언이 아니다. 위에서 언급한 인공지능, 사물인터넷, 확장현실 등 모든 것이 IT 기술이

다. IT 기술이 없다면 이러한 변화가 오지 않았을 것이다.

시가총액 TOP 10의 변화

기업의 가치는 일반적으로 시가총액(이하 시총)을 기준으로 판단하곤 한다. 이번 장에서는 최근 시총 상위 기업들이 어떤 유형의 기업들인지, 과거의 시총 상위 기업들 중 도태된 기업들은 어떤 유형의 기업들인지 등을 분석하여 최근 사회 변화와 이에 따른 기업 경영의 트렌드를 파악한다.

국내 기업의 변화

(표1)를 통해 지금으로부터 20년 전 국내 시가총액 상위 10개의 기업과 2020년 기준 상위 10개의 기업에 어떠한 변화가 있었는지 알아보자.

2000년 시총 2위였던 SK텔레콤은 2020년 15위로 내려갔고 3위였던 KT는 2020년 36위를 기록했다. 이 밖에도 포스코(2000년 6위 → 2020년 16위), LG유플러스(2000년 10위 → 2020년 40위) 등과 같이 상위 10위권에 있던 기업들의 순위가 급격한 하락세를 보이는 것을 확인할 수 있다.

반면 새로운 흐름도 감지되고 있다. 2010년 시총 25위였던 네이버가 4위를 기록했고 2017년에 상장한 카카오도 8위를 기록, IT 플랫폼 업

표 1 | 2000년 vs 2020년 시총 상위 기업 (단위: 억원)

2000년 시총 상위 종목				2020년 시총 상위 종목			
2000년 순위	종목명	시가 총액	2020년 순위	2020년 순위	종목명	시가 총액	2000년 순위
1위	삼성전자	55조6825	1위	1위	삼성전자	315조2045	1위
2위	SK텔레콤	32조5407	15위	2위	SK하이닉스	61조9530	5위
3위	KT	30조6580	36위	3위	삼성바이오로직스	51조2778	2016년 상장
4위	한국전력	22조1474	21위	4위	네이버	43조8583	25위
5위	SK하이닉스	10조7894	2위	5위	셀트리온	41조2914	2018년 상장
6위	포스코	9조1270	16위	6위	LG화학	34조6255	22위
7위	삼성전기	5조2210	27위	7위	삼성SDI	24조9959	25위
8위	KB금융지주	4조2545	17위	8위	카카오	23조4865	2017년 상장
9위	KT&G	3조8962	24위	9위	삼성물산	23조4865	32위
10위	LG유플러스	3조8667	40위	10위	LG생활건강	21조220	2001년 상장

※KT는 옛 한국통신공사, SK하이닉스는 옛 현대전자, 포스코는 옛 포항제철, KB금융지주는 옛 국민은행, KT&G는 옛 한국담배인삼공사, LG유플러스는 옛 데이콤

자료 : 한국거래소, 참고

계의 강세가 두드러지는 것을 확인할 수 있다. 이 밖에도 엔씨소프트 13위, 넷마블 29위(2000년 창업) 등 IT 기업이 대거 상위권으로 진입했다.

20년이 흐르는 동안 가장 큰 변화의 모습을 살펴보면 과거의 제조업 중심(자동차, 철강, 건설)의 산업이 뒤로 밀려나고 IT를 기반으로 중무장한 신유형의 산업이 빈자리를 메웠다. 20년 만에 국내 주력 산업의 패

러다임이 통째로 바뀐 셈이다. 최근 코로나19 사태로 인해 원격 근무와 같은 ‘언택트’ 문화가 빠르게 확산하고 있어 이러한 변화는 앞으로 더욱 가속화될 전망이다.

해외 기업의 변화

다음은 지금으로부터 20년 전 세계 시가총액 상위 10개의 기업과 2020년 3분기 기준 상위 10개의 기업에 어떠한 변화가 있었는지 알아보자(표 2 참조).











2000년대만 해도 가전제품 등을 생산하는 제너럴 일렉트릭(General Electric, GE)이 시총 1위를 기록했고, 엑손 모바일(Exxon Mobile)/보다폰(Vodafone) 등과 같은 통신회사가 각각 3위/8위를 기록한 것을 확인할 수 있다. 이 밖에도, 오프라인 기반의 쇼핑 체인인 월마트(Walmart), 컴퓨터 OS 및 각종 소프트웨어를 만드는 마이크로소프트(Microsoft) 등이 상위권에 올라와 있다.

2020년 3분기는 IT 기업의 세상이 되었다. 애플(Apple), 마이크로소프트(Microsoft), 아마존(Amazon), 알파벳(구글, Alphabet), 알리바바(Alibaba), 페이스북(Facebook), 텐센트(Tencent) 등의 IT 기업이 상위 7위까지를 휩쓸었다. 여기서 한 가지 더 주목할 점은 알리바바(Alibaba)와 텐센트(Tencent)와 같은 중국 IT 기업이 상위권에 당당하게 이름을 올린 것이다.

바야흐로 IT 기업의 전성시대가 온 것이다. 한국과 마찬가지로 20년 전에는 통신, 제조업 기반의 시장이었다면 20년이 흐른 지금은 IT 기

표 2 | 2000년 시총 상위

Rank	Name	Headquarters	Primary industry
1	General Electric	United States	Conglomerate
2	Cisco Systems	United States	Networking hardware
3	Exxon Mobil	United States	Oil and gas
4	Pfizer	United States	Health care
5	Microsoft	United States	Software industry
6	Wal-Mart	United States	Retail
7	Citigroup	United States	Banking
8	Vodafone	United Kingdom	Telecommunications
9	Intel Corporation	United States	Computer hardware
10	Royal Dutch Shell	The Netherlands	Oil and gas

Third quarter		
	Apple Inc.	▲1,981,000 ^[14]
	Microsoft	▲1,592,000 ^[13]
	Amazon.com	▲1,577,000 ^[15]
	Alphabet Inc.	▲999,570 ^[16]
	Alibaba Group	▲795,400 ^[17]
	Facebook, Inc.	▲746,100 ^[18]
	Tencent	▲646,790 ^[19]
	Berkshire Hathaway	▲509,470 ^[20]
	Visa Inc.	▲425,510 ^[21]
	TSMC	▲420,440 ^[23]

업 중심으로 모든 시장이 재편되었다. 앞으로 10년 후는 어떻게 될까? IT 기반으로 성장한 대기업의 성장세는 더욱 공고해질 것이며 후발 주자들도 무서운 속도로 따라갈 것이다. 그리고 미국의 주요 기업만이 성장하는 것이 아니라 중국의 IT 기업도 더욱 빠른 속도로 성장하고 있다는 사실을 잊으면 안 될 것이다.

IT 기업으로 변신은 무죄

앞서 언급한 바와 같이 전 세계 기업의 시총 순위가 완전히 뒤바뀌었고 이 변화의 핵심은 IT 기술과 개발자들이라는 말을 했다. 최근 국내외 여러 기업은 이러한 변화의 핵심을 빠르게 파악하고 기업 자체를 바꾸려는 움직임을 보이고 있다.

수십 년간 호황을 누렸던 금융 업계도 저성장, 저금리, 저물가라는 삼중고에 처해있다. 따라서 각국의 금융 분야 기업들은 생존을 위하여 다양한 변화를 시도 중이다.

골드만삭스는 금융회사 이미지를 버리고 기업 자체를 변화시키려고 다양한 노력을 하고 있다. IT 분야 인력을 대거 선발하고 핀테크 및 IT 기업 인수에 총력을 다하고 있다. 이처럼 글로벌 금융회사들은 기존 운영 방식에서 벗어나 인공지능 기반 고객 서비스 등 새로운 유형의 서비스를 개발하고 있다. 이는 기업의 생존과도 맞물려 있는 문제이므로 총력을 다하고 있다.

한국의 금융 업계는 더 심각한 상황이다. 국내 은행 이익의 약 88%가 이자 수익이라고 한다. 이자 수익 의존도가 높다는 것은 수익 구조가 금리 변동에 매우 민감하게 반응할 수밖에 없다는 것이다. 최근 저금리 기조가 장기간 지속되면서 국내 은행의 수익성은 계속 악화되고 있다. 게다가 인터넷은행인 카카오뱅크, 케이뱅크 등의 등장으로 이러한 수익성 악화는 더욱 심화되고 있는 상황이다. 실제로 4대 시중은행(신한, KB, 하나, 우리)이 최근 5년간(2015~2020년 8월 말) 폐점한 은행 점포 수는 549개로 나타났다. 앞으로도 기존 점포의 폐점 속도는 더욱 빨라질 전망이다. 이는 코로나19로 인한 비대면 문화의 확산과 인터넷·스마트폰 뱅킹을 활용하는 비율이 늘어났기 때문으로 분석된다. 이러한 상황에서 생존을 위해 국내 은행들은 IT 기반의 디지털 전환을 추진하고 있다. 우선 인공지능 기술을 접목하여 고객 상담 업무(챗봇)/신용도 평가/이상 거래 탐지 등을 시행하고 있으며 IT 자원을 더욱 유연하게 활용할 수 있는 클라우드 접목 추진, 고객 빅데이터 활용 맞춤형 서비스 제공 등 다양한 분야를 활용하여 디지털 전환을 추진하고 있다.

IT 기업으로 디지털 전환을 하려는 기업은 금융 업계뿐만 아니라 대부분 업종에서 일어나는 현상이다. 전통적인 제조업의 불량품 탐지 등을 위해서, 보다 과학적인 스마트 농업을 위해서, 정확하고 과학적인 물류 유통을 위해서는 반드시 IT 기반의 디지털 전환이 필요하다. 즉, 전통적인 산업 대부분에서 디지털 전환이 필요한 시기이며 지금 이 시기를 놓친다면 그 기업은 돌아올 수 없는 길을 걸을 수밖에 없다.

IT 기술의 핵심은 개발자

우리 회사가 기존의 전통 산업 기반으로 성장했던 기업이라면 이제는 IT 기반의 디지털 전환이 필요한 시기가 다가왔다. 시장은 매우 냉정해서 변화하지 않는 기업은 곧바로 도태되고 만다. IT 기반의 디지털 전환을 위해 반드시 필요한 존재는 누구일까? 필자는 이를 위해 가장 중요한 존재를 단연 개발자라고 단언할 수 있다. 은행권에서는 ‘행원 대신 개발자를 모셔라’라는 말이 나올 정도다. 최근 인터넷은행 3사(케이뱅크, 카카오뱅크, 토스뱅크)가 동시에 경력직 채용에 나섰다. 기존 은행에 비해 파격적인 혜택을 내세워 개발자를 모시기에 안간힘을 쓰고 있다. 엔터테인먼트 기업인 빅히트는 IT 인력을 대거 채용하면서 ‘IT 기반의 콘텐츠 기업’으로 탈바꿈했다. 경기도 판교 등지의 IT 기업에서 근무하는 개발자 100명 이상이 빅히트로 이직했다고 알려졌으며 지금도 다양한 분야에서 개발자를 채용하고 있다. 빅히트에서는 어렵게 채용한 개발자들을 활용해 다양한 디지털 상품을 출시했으며 최근에는 게임업체 넷마블과 협력해 BTS를 활용한 게임을 출시하기도 했다.

이처럼 새로운 서비스 개발 등을 위해 유능한 개발자를 채용하는 것은 필수적이며 이는 기업의 생존과도 직결되는 문제이기 때문에 반드시 고려해야 하는 시대가 되었다. 앞서 언급한 기술을 서비스로 구현하기 위해 단순히 아이디어만 가지고는 할 수 없다. 이를 눈에 보이는 서비스로 만들어 낼 기술자, 즉 개발자라는 존재가 필요하다. 과거 20년 사이에 산업의 패러다임이 IT 기반으로 변화하는데 가장 많은

기여를 한 직종은 무엇일까? 필자는 단연 개발자라고 말할 수 있다. 위에서 언급한 IT 기업들은 개발자 없이 생존할 수 없는 구조를 지니고 있으며 유능한 개발자를 뽑기 위해서 삼고초려의 노력을 통해 위와 같은 성과를 거두었다. 결과적으로 개발자가 없으면 기업의 생존도 위태로운 시대이며 이러한 변화는 점점 더 심해질 것이다.

여러분은 개발자라는 용어에 익숙하지 않을 수 있다. 개발자는 소프트웨어를 활용하여 제품과 서비스를 개발할 수 있는 기술자를 의미한다. 여러분이 지금 사용하고 있는 스마트폰, 가전제품, 자동차, 컴퓨터, 컴퓨터 프로그램 등 개발자의 손이 닿지 않는 곳이 없다고 보면 된다. 이렇게 중요한 역할을 하는 개발자가 당신 회사에 없다면 어떨까? 최근 불고 있는 IT 기반의 디지털 전환(Digital Transformation)을 제대로 수행할 수 없을 것이며, 이를 진행하더라도 실패할 확률이 높아질 것이다.

02

개발자 채용 전쟁

개발자 채용을 위한 소리 없는 전쟁

국내외의 많은 기업들이 IT 기반으로 디지털 변화를 시도하고 있다. 변화를 추구하는 기업들은 필수 인력인 IT 분야 혹은 소프트웨어 개발자 채용에 안간힘을 쏟고 있다. 기업에서 유능한 개발자를 모시기 위해 어떤 방법을 활용하는지, 이들이 만족하며 근무할 수 있는 환경을 어떻게 마련하는지 등을 여러 사례를 통해 알아본다.

업계 최고 수준의 연봉을 드립니다

잡코리아(Jobkorea)에서 2020년 4년 대졸 신입직 초임을 분석(총 771개 기업 조사)한 결과 대기업 대졸 신입의 초임은 평균 4,130만 원이며, 공기업 3,810만 원, 중소기업 2,800만 원 순으로 조사되었다.

하지만 개발자 채용이 매우 어려운 지금의 상황에서 위의 평균 연봉으로는 원하는 역량을 가진 개발자를 구할 수 없다. 따라서 스타트업

은 물론 IT 분야 대기업 등은 높은 연봉 및 복지 제공을 통해 유능한 개발자 선점에 안간힘을 쓰고 있다.

최근 개발자들 사이에 ‘네카라쿠배’라는 생소한 용어가 등장했다. ‘네카라쿠배’는 네이버, 카카오, 라인, 쿠팡, 배달의 민족을 의미하는데, 직장인 익명 커뮤니티를 중심으로 퍼지면서 신조어로 자리 잡았다. ‘네카라쿠배’는 국내 IT 분야를 주도하고 있는 기업들이다. 플랫폼 기반의 다양한 서비스를 통해 가파른 확장세를 보이고, 이러한 성장세에 따라 이들 기업은 개발자들이 가장 입사하고 싶어 하는 회사가 되었다. 이들 기업은 최고 수준의 연봉과 다양한 복지 혜택, 무한한 성장성을 무기로 최고 수준의 개발자를 채용하고 있다. ‘네카라쿠배’ 기업들의 개발자 신입 연봉은 4,000만 원~5,000만 원 수준으로 대기업 신입 초임보다 비교적 높은 수준이다. 경력직의 경우 이들이 가진 역량에 따라 천차만별이라고 한다.

IT 분야 대기업인 ‘네카라쿠배’에서 공격적인 인재 유치에 나서면서 중견 IT 기업과 스타트업은 인재 유치에 비상이 걸렸다. 실제로 최근 중견 IT 기업의 유능한 개발자들이 ‘네카라쿠배’로 대규모 이직을 했다고 한다. 따라서 상대적으로 불리한 위치에 있는 기업들은 유능한 개발자를 대기업에 빼앗기지 않기 위해 더욱 높은 연봉과 다양한 복지 혜택을 통해 생존 전략을 마련하고 있다.

토스는 경력직 입사자에게 이전 회사 연봉의 1.5배를 제안하는 것으로 알려졌다. 여성 쇼핑몰 모음 서비스인 지그재그는 신규 입사 경력자에게 전 직장 연봉 대비 30% 인상을 약속해준다. 중고 거래 플랫폼

인 당근마켓은 400억 원의 투자를 유치한 후 개발자의 연봉은 최소 5,000만 원을 보장한다는 채용 공고를 냈다. 대부분 중견기업/스타트업은 개발자의 최저 연봉 5,000만 원 수준을 보장하고 있으며 이보다 더욱 많은 금액을 지급하는 회사들도 늘어나는 추세다.

입사 시 무조건 보너스 5천만 원을 드립니다

IT 분야 개발자 찾기가 하늘의 별 따기 수준으로 힘든 상황이므로 많은 회사들이 파격적인 조건을 내걸고 공격적인 인재 영입에 나서고 있다. 그중 주목할만한 조건으로 '사이닝 보너스(Signing Bonus, 입사 축하금)' 제도를 들 수 있다. 실제로 2000년대 초반 삼성, LG 등은 사이닝 보너스 제도를 통해 유능한 인재를 채용했다. 미국 메이저리그는 지금도 이러한 제도를 잘 활용하고 있다. 최근 개발자 채용이 매우 어려워지면서 국내 스타트업도 사이닝 보너스 제도를 도입하는 기업들이 하나둘 늘어나고 있다. 사이닝 보너스란 회사에 새로 합류하는 직원에게 주는 일회성 인센티브를 말한다. 일반적으로 사이닝 보너스를 받은 직원은 몇 년간 다른 회사로 이직할 수 없으며 만약 이직할 경우 이미 받은 보너스를 반환하는 조건 등이 붙는다. 따라서 해당 보너스 지급을 통해 우수한 역량을 가진 개발자를 채용하고 이를 장기간 유지하고 싶어 하는 것으로 파악된다.

최근 쿠팡은 약 200여 명의 기술 직군 경력 공채를 시행했는데 합격자들에게 최소 5천만 원의 사이닝 보너스를 지급한다고 전해졌다. 물류 스타트업 메쉬코리아는 AI, 데이터 사이언스, 엔지니어링 등의 개발

자 경력직 공채를 시행하면서 최소 5,000만 원에서 많게는 1억 원 정도의 사이닝 보너스를 지급한다고 알려졌다. 모바일 금융 서비스 기업인 토스도 최근 파격적인 보너스 제도를 도입했다. 입사 시 이전 회사 연봉에 준하는 금액(최대 1억 원)을 사이닝 보너스 혹은 스톡옵션으로 제공한다. 사이닝 보너스는 입사 후 첫 월급일에 일시금으로 지급한다고 알려졌다.

그렇다면 이렇게까지 하는 이유는 무엇일까? 최근 4차 산업 기술 분야의 인력이 매우 부족한 상황이고 이 분야의 핵심 인력이 바로 개발자이기 때문이다. 스타트업은 안정적인 서비스를 빠르게 개발해야 하므로 타 경쟁업체보다 유능한 개발자를 유치하는 것이 가장 중요하다. 이러한 보너스 제도 도입을 통해서라도 우수한 개발자를 모셔야 경쟁력 있는 서비스 개발이 가능하기 때문이다.

최고의 근무 환경과 복지를 제공합니다

개발자들은 다른 직군에 비해 더욱 자유로운 근무 환경을 추구한다. 본인이 가장 근무하기 편한 곳에서 작업해야 업무의 효율성이 높아지기 때문이다. 따라서 집, 회사 등 어디든 본인이 편한 곳에서 개발을 할 수 있도록 배려해주는 기업들이 점차 늘어나고 있다. 이외에도 특별한 휴가 제도, 대출 제도 등 다양한 복지 혜택을 통해 유능한 개발자의 장기근속을 장려하는 기업들이 늘어나고 있다.

NHN의 기술 전문 법인인 NHN 토스트는 주 8시간의 오피스 근무를 제외하고 나머지 시간에는 전면 재택 근무가 가능한 ‘오피스 프리’

제도를 도입하여 운영한다. 또한 근속 연수에 따라 안식 휴가를 제공하는데 3년 근속 시 1개월의 유급휴가가 주어지고 5년 근속 시 2개월, 10년 근속 시 6개월간 유급휴가가 주어진다.

우아한 형제들의 직원들은 매주 월요일엔 오후에 출근한다. 직장인들이 가장 심리적으로 힘들어하는 월요일 오전을 자유롭게 보내고 출근할 수 있어 직원들의 만족도가 매우 높다고 한다. 직원들의 만족도 뿐만 아니라 업무의 효율성도 높아졌다고 하여 이를 벤치마킹하는 기업들도 늘어나는 추세다. 또한, 주 35시간 근무를 하고 1년 이상 근속한 직원들에게 '2주 리프레시 휴가'를 제공한다. 이 밖에도 임산부 단축 근무제, 1개월간 특별 육아 휴가(유급), 학부모 특별 휴가 등 다양한 복지제도로 직원들의 만족도 향상을 위해 노력하고 있다.

당근마켓은 개인 사정에 맞춰 출/퇴근 시간을 조절할 수 있는 자율 출퇴근 제도를 활용하고 있으며 매주 목요일은 재택 근무를 통해 업무의 효율성을 높이고 있다. 또한, 법인 카드로 회사 내 필요한 물품 구매, 간식, 식사 등을 자유롭게 결제할 수 있어서 직원들의 만족도가 높아졌다고 한다. 휴가 일수도 제한 없이 활용할 수 있어 필요한 경우 언제든지 쉴 수 있는 환경을 구축했다.

토스는 출퇴근 시간을 자유롭게 조절하는 자율 출퇴근제, 사용 횟수 등 제한이 없는 원격 근무제, 승인이 필요 없는 휴가 무제한 사용 등 파격적인 근무 환경을 제공한다. 또한, 직원 개인에게 법인 카드를 제공하고 1억 원의 주택 자금을 무이자로 대출해준다. 직원들의 체력 관리를 위해 매일 체력 단련비를 제공하는 것도 큰 특징 중 하나다. 토

스는 이러한 근무 환경 도입으로 신규 입사자의 근속 비율이 약 90%에 달한다고 한다.

넥슨의 자회사인 게임 업체 '네오플'은 본사가 제주도에 위치한다는 핸디캡을 극복하고자 타지에서 온 신규 직원에게는 아파트를 사택으로 제공한다. 미혼자는 89㎡(약 27평), 기혼자는 105㎡(약 32평) 규모의 아파트를 제공하여 거주지 이전에 대한 부담을 덜어주었다.

이 밖에도 취미 공유 플랫폼인 클래스 101은 장거리 거주자에게 사택을 제공하며 월세는 회사에서 전액 부담한다. 또한, 유연근무제 도입, 외근 및 야근 시 교통비 지원, 독서 비용 무제한 지원 등의 다양한 복지제도를 도입하여 운영하고 있다.

앞서 언급한 것과 같이 대부분의 IT 기업은 개발자가 최고의 환경에서 근무할 수 있도록 다양한 복지제도를 운영하고 있다. 이는 결과적으로 개발자가 회사에 만족하면서 다닐 수 있는 큰 요인 중 하나로 작용한다. 개발자가 회사에 만족하면 이는 업무의 효율성과도 연계되므로 기업 입장에서는 더욱 많이 신경을 써야 하는 부분 중 하나이다.

개발자에 대한 인식 변화

불과 몇 년 전까지만 해도 개발자라고 하면 3D 업종이라고 생각한 사람들도 있었다. 하지만 불과 몇 년 사이 이러한 인식은 많이 개선되었다. 그동안 사회적으로 다양한 변화가 있었는데 아래에 설명한 요인

에 기인한 것으로 파악된다.

● 대학 입시 결과의 변화

80~90년대 초반의 인기 있는 학과중 하나가 '컴퓨터공학'였다. 컴퓨터공학과는 상위권 성적의 학생들이 선호하는 학과였다. 이는 실제 입시 결과에서도 나타났는데 1993년 대성학원 학력고사 점수별 대학 입학 배치기준표의 가장 앞쪽에는 서울대학교 컴퓨터공학과, 물리학과 등이 차지했다. 그로부터 약 20년이 흐른 후 2010년 대성학원 대입 배치표에 따르면 백분위 점수 상위 50위 중 서울대 생명공학부, 자유전공학부를 제외하고는 모두 의예, 한의예 학과가 나머지를 차지했다. 불과 20년 만에 의예, 한의예 관련 학과가 강세를 보였다.

그렇다면 왜 이러한 결과가 나온 것일까? 이는 한국에서 소프트웨어 관련 분야로 진출할 경우 잦은 야근과 과중한 업무량, 낮은 연봉 등으로 인하여 좋은 대우를 받지 못한다는 인식이 널리 퍼졌기 때문이다. 실제로 SI 분야 개발자들의 이야기를 들어보면 아직도 워라벨이 잘 지켜지지 않는다는 곳이 많다. 이는 좋은 인력들이 오랫동안 근무할 수 있는 여건이 되지 않는다는 것이다. 이에 피로감을 느낀 개발자들이 다른 분야로 이직을 하거나 전공을 바꿔 대학원 진학 등을 통해 직종의 전환을 하는 경우도 많았다.

하지만 다시 시대가 달라졌다. 그로부터 약 10년이 흐른 2019년, 컴퓨터공학과가 다시 높은 인기를 회복했다. 약 30년 만에 최상위권 학생들이 원하는 학과가 되었고 이는 입시 결과에도 그대로 반영되었다.

2019년 서울대 정시에서 컴퓨터공학과 합격선은 406점(수능 표준점수 600점 기준)이었으며 의예과는 411.4점, 치의학과는 406.6점이었다. 의예과를 선호하는 현상은 비슷하지만 이제 컴퓨터공학과가 이에 버금가는 인기를 얻는 과라는 것을 실감할 수 있는 부분이다.

● 대학의 변화

최근 과학기술정보통신부에서 지원하는 소프트웨어 중심대학 사업의 규모와 지원 비용을 보면 소프트웨어 분야의 인기를 다시 한번 실감할 수 있다. 소프트웨어 중심대학은 대학교육을 소프트웨어 중심으로 혁신함으로써, 소프트웨어 전문인력을 양성하고 학생, 기업, 사회의 소프트웨어 경쟁력을 강화하여 진정한 소프트웨어 가치 확산을 실현하는 대학을 말한다. 소프트웨어 중심대학은 2015년부터 사업이 시작되어 2019년 하반기(10월)까지 총 40개 학교가 선정되었다. 학교당 연간 10~20억 원의 예산이 지원된다.

이 예산을 통해 대학 교육과정을 소프트웨어/AI 중심으로 개편, 융합 학과 개설, 해당 분야 교원 초빙, 산학협력체계 구축 등을 수행하게 된다. 실제로 소프트웨어 중심대학 사업에 선정된 대학은 신입생 전원을 대상으로 입학 전/후 소프트웨어 교육 실시(파이썬), 소프트웨어/AI 융합 학과 신설, 컴퓨팅 사고력 기반의 소프트웨어 소양 교육 등을 실시하여 소프트웨어 분야의 인력 양성을 위해 다양한 노력을 수행하고 있다.

● 초·중등 소프트웨어 교육의 의무화 및 인기 상승

초·중등 교육에서도 소프트웨어 교육의 인기가 날로 상승하고 있다. 일반적으로 우리가 학창시절부터 생각해왔던 주요과목은 국어와 영어, 수학 정도였다.

하지만 최근 소프트웨어 교육이 중요해지면서 초·중등 과목 내 반드시 배워야 하는 필수과목으로 자리 잡았다. 관련 학과의 인기도 덩달아 상승하면서 주요과목이 ‘국영수코’(국어, 영어, 수학, 코딩)라는 말이 나올 정도이다.

이에 따라 소프트웨어 교육 관련 학원 등의 사교육 시장도 빠르게 성장하고 있다. 이는 학생과 학부모의 관심을 대변하는 것으로 생각된다. 또한, 초·중등 학생 대상으로 소프트웨어 교육 및 컴퓨팅 사고 등의 교육을 수행할 수 있는 언플러그드, 로봇 코딩, 블록 코딩, 피지컬 컴퓨팅 등의 소프트웨어 교육 도구들이 출시됐다.

● 인공지능에 대한 일반인의 관심 증대

2019년 12월 정부는 ‘인공지능(AI) 국가전략’을 발표를 통해 AI 강국으로 발돋움하기 위한 초석을 마련하였다. 해당 전략에서는 3대 분야의 9대 전략을 수립하고 100대 실행과제를 제시하였다. 이를 통해 AI 분야 인재양성을 위해 초급 인재부터 최고급 인재까지 양성하는 것을 목표로 하고 이를 바탕으로 다양한 부처에서 구체적인 전략을 실행하고 있다.

해당 전략은 전 국민이 인공지능 기본 소양을 갖게 하는 내용도 포

함되었는데, 전 생애 동안 모든 국민이 인공지능을 잘 활용할 수 있게 만드는 것을 목적으로 한다.

이러한 정부의 정책과 더불어 여러 기업에서 다양한 형태의 인공지능 관련 제품을 출시하면서 대중들의 관심도 빠르게 늘어나고 있다. 요즘에는 유치원생도 인공지능이라는 단어를 들어봤을 정도로 인공지능이라는 용어가 많은 곳에서 사용되고 있다. 이는 해당 분야의 서비스 개발에 대한 요구가 많아지는 것으로 이에 따라 관련 분야의 개발자 채용으로 이어지고 있다.

소프트웨어정책연구소에 따르면 실제로 인공지능 분야 연구 및 개발 인력은 현재도 많이 부족하며 2023년까지 약 3만 명의 AI-소프트웨어 분야 인력이 부족할 것으로 전망했다.

● 시대별 개발자에 대한 인식의 변화

요즘 개발자들을 만나보면 예전보다 개발자에 대한 처우가 많이 좋아졌다고 한다. 대부분 높은 수준의 연봉을 받고 있으며 근무 환경도 좋아졌다는 말을 한다. 그렇다면 개발자들은 원래부터 이러한 처우를 받았을까?

해당 내용을 구체적으로 확인하기 위해 90년대부터 최근까지의 흐름을 살펴보겠다.

1990년대에는 소프트웨어 시장이 별도로 존재하지 않았던 시대였다. 이때 대부분의 개발자는 취미 삼아 개발을 진행하는 정도였으나 실력은 뛰어난 경우가 많았다.

2000년대에는 학원 등에서 교육을 받은 개발자들이 다수 배출됐다. 이때는 SI 분야의 성장으로 인해 개발자들의 수요가 급증한 시기였다. 하지만 이러한 수요에 비해 개발자에 대한 처우 및 근무 환경은 좋지 않았다. 또한, 이때 채용된 개발자들은 비교적 만족스럽지 못한 개발 결과를 보여주었다. 이로 인해 개발자들에 대한 인식과 대우가 점점 안 좋아지는 악순환이 반복되었다.

2010년대는 모바일과 소셜 미디어가 폭발적인 성장을 한 시기이다. 따라서 관련 서비스를 개발하는 스타트업이 늘어났다. 관련 인력 수요는 급증하였고 이에 따라 개발자에 대한 처우도 자연스럽게 높아지는 계기가 되었다. 또한, 비교적 높은 경력을 가진 유능한 개발자들에 대한 수요도 동시에 증가하였으며 이러한 체계는 개발 업계의 선순환 체계를 가져오게 했다.

2020년대가 되면서 이러한 선순환 체계가 공고화되는 것으로 보인다. 유능한 개발자일수록 더 좋은 대우를 받고 좋은 환경에서 근무하는 비율이 높아졌다. 대기업뿐만 아니라 스타트업도 좋은 개발자를 뽑기 위해 삼고초려를 하는 경우도 많아졌다. 개발자들은 점점 본인이 원하는 수준의 회사를 골라서 갈 수 있는 환경이 되고 있다.

위에서 언급한 여러 요소가 복합적으로 작용하여 개발자들이 좋은 대우를 받는 환경이 마련되었다. 앞으로도 실력 있는 개발자를 필요로 하는 기업들은 더욱 빠르게 늘어날 전망이다.

주요 IT 기업 개발자 채용 현황

최근 주요 IT 기업의 채용 현황을 살펴보면 아래의 표와 같다. 다양한 분야에 대한 채용을 진행하고 있지만, 특히 개발자에 대한 채용 비율이 매우 높은 것을 파악할 수 있다. 표에서 제시하는 채용 공고 중에 개발자 채용을 파악할 수 있는 공식적인 통계 수치는 없다.

따라서 관련 채용 공고를 직접 살펴보고 개발자 관련 채용 공고의 비율을 파악했다. 해당 채용 공고는 2020년 10월 5일 기준으로 지원 가능한 채용 공고만을 대상으로 분석하였으므로 실제 전체 기업에서 채용하는 개발자 채용 비율과는 다소 다를 수 있다. 하지만 이를 통해 확인할 수 있는 핵심은 전체 채용 인원 중 개발자가 차지하는 비율이 상당히 높다는 점이다. 대부분 기업에서 절반 이상의 인력을 개발자로 뽑고 있으며 이를 통해 전체 채용의 경향을 추정할 수 있다.(표3 참조)

스타트업의 개발자 채용 비율

주요 IT 기업에서 많은 인력을 공격적으로 채용하고 있는 상황에서 스타트업이라고 가만히 있을 수는 없다. 아니, 가만히 있으면 안 된다. IT 분야 중견/대기업 규모의 기업은 기존 개발자들이 다수 있는 상황에서 추가적인 인력이 필요하여 채용하는 경우가 많기 때문에 스타트업과는 상황이 다소 다를 수 있다. 스타트업을 직원 수에 따라 분류하

표3 | 주요 IT 기업의 개발자 채용 비율

기업명(가나다순)	개발 직군 채용 비율
네이버	83.20%
넥슨	57.00%
넷마블게임즈	71.40%
비바리퍼블리카(토스)	46.40%
엔씨소프트	86.10%
우아한형제들	38.40%
위메프	47.60%
카카오	65.60%
크래프톤	48.00%
NHN 엔터테인먼트	57.40%

면 20인 미만인 기업과 20~50인 미만의 기업, 50인 이상의 기업 정도로 구분할 수 있다. 20인 미만인 기업은 초기 스타트업일 확률이 높으며 사업 경험이 많이 없고 서비스를 개발하기 위한 개발자가 충분하지 않을 확률이 높다. 직원 수 20~50인 정도의 스타트업은 이미 서비스를 출시한 상황에서 신규 서비스 확장, 기존 서비스 유지 보수 등의 인력이 필요할 것이다. 마지막으로 직원 수 50인 이상의 스타트업은 시장의 인지도가 어느 정도 쌓인 기업으로 안정적인 매출이 나오고 있을 확률이 높다. 해당 기업에서도 개발자가 필요하긴 마찬가지이며 초기 스타트업 때보다 경험이 많은 유능한 개발자를 뽑을 확률이 높다.

다음은 주요 스타트업의 채용 규모(2018년)를 나타낸 표이다. 이 표

에서 주목할 것은 스타트업 채용 규모 중 개발 직군이 차지하는 비율이 대부분 50%를 선회한다는 점이다. 앞서 살펴보았던 주요 IT 기업의 수치와 어느 정도 일치하는 부분이다. 그만큼 스타트업에서도 개발자가 아주 중요한 역할을 한다는 것을 유추할 수 있다.

표4 | 스타트업 채용 규모

회사	대표 서비스	채용 규모	개발직군
우아한형제들	배달의민족	400명	200명
위드이노베이션	여기어때	200명	100명
야놀자	야놀자	200명	100명
알지피코리아	배달통·요기요	150명	50명

03

개발자가 ‘갑’ 이다

회사에서 채용을 진행할 때 직원이 갑이라고 생각해본 적이 있는가? 대부분 그렇게 생각하지 않을 것이다. 하지만 개발자의 세계는 많이 다르다. 개발자 입장에서 보면 유능한 개발자는 갈 곳이 많고 우리 회사에 꼭 올 필요가 없다. 이렇게 고민하는 개발자를 우리 회사로 ‘잘’ 모셔와야 한다. 우리 회사에서 월급을 주니 회사가 갑이라고 생각하면 개발자 채용에서 큰 실패를 맛보게 될 것이다. 다시 한번 말하지만 개발자가 ‘갑’이고 회사가 ‘을’이다. 개발자들은 입사 후 회사가 마음에 들지 않으면 언제든지 이직할 준비가 되어있다.

잠시 머물다 가는 개발자들

우리 회사에서 원하는 역량을 개발자를 뽑았다고 해서 좋아할 때가 아니다. 이 개발자는 언제든지 떠날 준비를 하고 있음을 잊지 말아라.

따라서 어렵게 채용한 개발자가 회사를 떠나지 않도록 하는 것도 중요한 일이다.

개발자들의 이직은 조직의 크기와 관계없다

회사의 조직이 매우 큰 경우와 반대로 매우 작은 경우, 둘 중 어떤 경우에 개발자들의 이직률이 높을까? 명확한 정답은 없지만 몇 가지 사례를 통해 이들의 이직은 기업 규모와 관계없다는 것을 설명하려고 한다.

미국 실리콘밸리에 있는 IT 기업들은 무료 식사 및 간식 제공, 교통편 제공, 자유로운 근무 환경, 주거 제공 등의 화려한 복지로 직원들을 유혹한다. 링크드인 자료에 따르면 실리콘밸리 직원의 근속 연수는 우버 1.8년, 드롭박스 2.1년, 테슬라 2.1년, 페이스북 2.5년, 에어비앤비 2.6년으로 나타났다. 남들이 보면 매우 부러운 수준의 연봉과 복지를 받고 있음에도 불구하고 이렇게 짧은 근속 기간을 보이는 이유는 무엇일까? 이렇게 큰 규모의 기업인데 2년 정도밖에 근무를 못 하는 이유는 무엇일까?

IT 분야는 다른 분야보다 이직률이 매우 높은 분야이다. 괜찮은 사람을 못 뽑아도 걱정, 뽑아도 걱정인 것이다. 유능한 개발자의 경우 회사를 나와도 갈 곳이 많다. 지금의 직장을 계속 다니는 이유는 연봉과 복지가 만족스럽고 다양한 경험을 통해 본인이 성장할 수 있는 가능성이 높은 곳일 수 있다. 여기에서 이직을 위한 가장 중요한 요소 세 가지를 뽑아내면 '연봉'과 '성장 가능성', '조직 문화'이다. 이 요소 중에

개인의 가치를 어디에 두고 있는지에 따라 다르겠지만 개발자의 대부분은 세 가지 요소 중 하나가 만족스럽지 않으면 떠날 준비를 한다. 만약 모두 불만족스럽다면 그 개발자는 회사에 이미 없을 것이다. 그만큼 개발자들은 언제든지 이동할 수 있는 준비가 된 사람들이며 이들을 잘 붙잡아 둘 수 있는 유인책이 있어야 한다.

당신의 기업이 작은 스타트업이라고 기죽을 필요는 없다. 앞에서 언급한 것처럼 우수한 대기업에서도 2년이면 다 나간다. 만약 '성장 가능성'을 가장 큰 가치로 두고 있는 개발자라면 스타트업에서 새로운 서비스를 개발하며 다양한 경험을 쌓기 원할 것이다. 여기에 연봉까지 이전 회사보다 높게 줄 수 있다면 금상첨화이다. 이러한 개발자가 있다면 절대 연봉을 아끼지 말고 채용해서 우리 회사의 핵심 인력으로 만들고 성장시키는 것이 좋다. 앞서 언급한 것처럼 '조직 문화'도 이직의 아주 중요한 요소이다.

'이 회사는 나를 소모품 취급한다', '야근을 밥 먹듯이 한다', '회사가 추구하는 가치와 잘 맞지 않는다' 등 조직 문화가 개발자 본인과 맞지 않는다면 뒤도 돌아보지 않고 떠날 확률이 높다.

잡은 이직은 개발자 채용의 숙명이니 받아들여야 한다. 다만, 어렵게 채용한 개발자가 한 달 만에 퇴사한다면 우리 회사에 문제가 있는 것이니 해당 문제가 발생하지 않도록 조직 문화, 개발 환경, 복지 등 다양한 부분에 미리 신경을 쓰고 준비해야 한다. 이러한 준비가 잘된 상황에서 개발자를 채용한다면 실패 확률을 훨씬 줄일 수 있다.

개발자가 없으면 스타트업도 없다

스타트업의 CEO들이 가장 두려워하는 것이 개발자의 이직이라고 한다. 스타트업에서 제대로 된 개발자를 뽑는 것 자체가 매우 어려운 일이며, 운이 좋아서 이러한 개발자를 뽑았다고 해도 안심할 수 없다. 만약 서비스 전체를 총괄하고 있는 핵심 개발자가 다른 기업으로 이직하게 되면 사업 자체가 중단되는 난처한 상황이 발생한다. 개발자 출신 CEO라면 본인이 어느 정도 진두지휘해서 수습은 하겠지만 대부분의 CEO는 개발자 출신이 아니므로 상당한 어려움이 발생한다. 해결을 위해 단기간 내에 유사한 역량을 가진 개발자를 채용해야 하지만 이게 말처럼 쉬운 게 아니다.

다음 문제는 스타트업 초기 단계에서 외부 투자를 유치하는 과정에서 발생한다. 정말 기발한 사업 아이디어를 가지고 있으나 이를 구현할 개발자가 없다면 투자를 받기 쉽지 않다. 투자 유치 면접에서 ‘우리 기업은 투자를 받은 후 개발자를 구해서 개발하겠다’라고 말하면 대부분 탈락의 고배를 마실 수밖에 없다. 이렇듯 스타트업의 개발자는 스타트업의 생존을 위해 반드시 필요하며 이들이 없다면 스타트업의 존재도 없는 것이다. 따라서 유능한 개발자를 ‘잘’ 채용해야 하며, 채용 후에도 떠나지 않도록 ‘잘’ 보살펴야 한다.

MZ세대 개발자들을 이해해야 한다

MZ세대라는 단어를 아는가? MZ세대란 밀레니얼세대(1980~2004년생)와 Z세대(1995~2004년생)를 통칭하는 신조어다. 이들은 전체 인구

의 약 33%, 주요 기업 구성원의 60%를 차지할 만큼 영향력이 커지고 있으며 생산과 소비 능력이 가장 좋은 주력 세대로 떠오르고 있다. 이들은 이전 세대와는 다른 성향을 가지고 있어 기업은 이들과 소통하기 위한 다양한 방법을 모색하고 있다. 당신이 MZ세대 직원들을 보며 “재들은 왜 저러는 거야?”라고 생각하고 있다면 좋은 CEO의 자격을 갖고 있지 않을 확률이 높다.

이들의 가장 큰 특징은 ‘자신이 지속적으로 성장할 수 있는지’에 큰 가치를 두는 점이다. 사람인에서 2030세대(MZ세대) 성인남녀 2,816명을 조사한 결과 약 절반 정도(49.4%)의 응답에서 ‘잡호핑족이 되고 싶다’는 설문 결과가 나왔다. ‘잡호핑족’이란 본인의 커리어를 위해 2~3년에 한 번씩 직장을 옮기는 사람을 의미하는 신조어다. 이러한 조사 결과는 많은 MZ세대들이 안정적인 직장보다는 다양한 경험을 위해 언제든지 움직일 준비가 되어있다는 것을 의미한다. 특히, 잡호핑족을 선호하는 이유 중 1순위가 ‘경쟁력 있는 커리어’를 위해서로 나타나 이들이 생각하는 직업에 대한 가치를 엿볼 수 있다.

개발자 세계에서도 MZ세대가 핵심적인 역할을 담당하고 있다. 당신의 회사에서 개발자를 뽑았다면 대부분 MZ세대일 것이기 때문에 이들에 대한 이해가 반드시 필요하다. ‘나 때는(라떼는) 말이야’라는 말과 함께 등장한 신조어인 ‘라떼는’은 MZ세대에게는 통하지 않는다. 내가 신입 사원일 때는 잦은 야근, 주말 근무, 잦은 회식 참석을 했다고 해서 이들에게 이러한 문화를 강요하면 안 된다. MZ세대는 효율성과 공정성을 중시하는 세대로 이직률이 매우 높다. 회사에 대한 충성심보

다는 개인의 삶을 더욱 중시하고 불필요한 회의/회식 등 효율성이 낮은 일에 대해 극히 거부감을 보이는 등 이전 세대와는 다른 특징을 보인다. 따라서 회사에서는 이들의 문화를 이해하고 공감하는 과정이 필요하다. 또한, 이들의 특성에 맞는 근무 환경, 성장 가능성을 만들어 줘야 장기 근속할 확률이 높아진다. 만약 이들을 이해하지 못하면 입사 후 한 달 만에 소중한 개발자가 떠나버릴지도 모른다.

항상 채용하는 마음가짐을 가져라

채용 공고를 내고 기다리면 역량 있는 개발자가 우리 회사에 지원할까? 물론 그럴 확률도 있지만 그렇지 않을 확률이 더 높다. 채용 공고를 내는 것만이 채용이 아니다. 수시로 유능한 개발자가 나타나면 채용할 준비가 되어있어야 한다. 이를 위해서는 몇 가지 준비해야 할 것들이 있다.

채용은 한 번에 이루어지는 것이 아니다

채용 공고 한 번으로 유능한 개발자를 뽑을 수 있다고 생각하면 절대 안 된다. 네이버나 카카오에서 개발자 채용 공고를 내면 많은 개발자들이 줄을 서서 지원할 것 같은가? 이렇게 큰 규모의 기업에서도 유능한 개발자를 뽑는데 상당한 어려움을 호소하고 있다. 그렇다면 효율적인 채용을 위해서는 어떠한 노력을 하면 좋을까? 우선 다양한 분

야에서 일하고 있는 개발자들과 꾸준히 네트워킹을 해야 한다. 특히 스타트업이라면 이러한 과정이 더욱 중요하다. 본인의 인맥을 찾다 보면 잠재력이 풍부한 개발자들 혹은 이들을 추천해줄 만한 사람들이 분명히 있을 것이다. 이들과 꾸준한 소통 관계를 유지할 필요가 있다. 지금 당장은 이들과의 관계가 별로 도움이 되지 않을 것처럼 생각되겠지만, 언젠가는 반드시 이들이 우리 회사에 도움이 되는 일이 있을 것이다. 만약 유능한 개발자가 있다면, 우리 회사에서 뽑는 포지션과 일치하지 않는 경우라도 이들과의 관계를 소홀히 하면 안 된다. 분명 우리 회사에서 그 개발자가 원하는 포지션을 뽑을 때가 있을 것이며 이때를 기회로 활용하면 되는 것이다. 유능한 인재는 한 번에 뽑을 수 있는 것이 아니고 꾸준한 준비와 네트워킹을 통해서 만들어어나가는 것이다.

적극적인 구매와 이를 위한 준비가 필요하다

만약 주변에 우리 회사에 꼭 필요한 역량을 가진 개발자가 있다면 회사로 모셔올 수 있는 좋은 기회이다. 하지만 보통의 노력으로는 이 개발자를 채용하기 힘들 것이다. 앞서 언급한 것처럼 꾸준히 접근하는 노력이 필요하다.

개발자 세계에서는 '삼고초려'의 마음가짐으로 개발자를 뽑아야 한다. 우리 회사에 매우 적합한 개발자를 발견했다면 이제부터 게임 시작이다. 당신이 CEO이든 HR 부서장이든 직함은 상관없다. 이들을 모시기 위해 자존심을 최대한 버리고 낮은 자세로 접근해야 한다. 네이버, 카카오 등의 IT 대기업들도 삼고초려의 마음가짐으로 개발자를 채

용하고 있으며 이들이 최대한 좋은 환경에서 근무할 수 있도록 채용 후에도 많은 신경을 쓴다. 하지만 우리 기업은 위에서 언급한 규모의 기업이 아니기에 더욱 낮은 자세로 채용에 임해야 한다. 모든 개발자가 대기업을 선호하는 것은 아니므로 우리 회사에도 기회는 있다.

대신에 이를 위한 다양한 무기가 필요한데 이 무기 중 첫 번째가 정성이다. 삼고초려의 진심을 담은 마음가짐으로 개발자와 소통할 수 있는 정성이 있어야 한다. 이러한 노력만이 개발자에게 감동을 줄 수 있다.

우리 회사가 스타트업이라면 더 많은 정성을 쏟아야 한다. 요즘은 스타트업이 개발자를 선택하는 것이 아니고 개발자가 가고 싶은 스타트업을 선택하는 시대다. 초기의 스타트업은 아직 서비스가 개발되지 않은 상황이기 때문에 개발자를 설득하기 어려운 상황일 수 있다. 이런 경우에 정성만 가지고 있다고 해서 개발자를 채용할 수는 없다. 추가로 우리 회사에 대한 정보를 최대한 구체적으로 준비해야 한다. 즉, 우리 회사 서비스 개발을 통해 어떤 성장을 할 수 있을지에 대한 확신을 심어줘야 한다. 하지만 가시적으로 보여줄 수 있는 것에는 한계가 있으므로 설득이 쉽지 않을 것이다. 따라서 개발자를 설득하기 가장 어려운 초기 스타트업의 경우 구체적인 성장 가능성, 성공 경험, 새로운 기술 스택에 대한 경험, 편안한 협업이 가능한 동료들, 쾌적한 근무 환경, 확실한 보상 및 성과급 체계 등을 구체적으로 준비하여 우리 회사가 매우 매력적인 기업이라는 것을 어필해야 한다. 준비하지 않으면 좋은 기회가 왔을 때 놓치는 순간이 올 수밖에 없다. 이렇게 철저하게 준비하고 다양한 인맥을 통해 우리 회사에 올 개발자를 적극적으로

찾아라. 그러면 우리 회사에 꼭 필요한 개발자를 영입할 확률이 몇 배는 올라갈 수 있다.

2020년 채용 트렌드

최근 채용 트렌드는 어떨까? 단순히 채용 공고를 내고 지원자를 기다린다면 제대로 된 인재를 구할 수 있을까? 시대가 바뀌었다. 이렇게 가만히 있을 때가 아니다. 우리 회사에 필요한 인재를 적극적으로 찾아 나서야 한다. 몇 년 전과는 다르게 최근에는 채용 트렌드 자체가 급변하고 있다. 이번 장에서는 이러한 변화의 흐름을 알아본다.

기업들이 인하우스 리크루터에 주목하기 시작했다

최근 5년간 인하우스 리크루터(사내 채용담당자)를 활용한 채용이 3배나 늘었다고 한다. 그만큼 사내에서 인하우스 리크루터의 역할이 중요해지고 있다, 예전에는 헤드헌터를 활용한 채용이 많았다면 요즘은 회사에서 직접 인하우스 리크루터를 채용하는 방식으로 트렌드가 변화하고 있다.

그렇다면 인하우스 리크루터는 어떤 업무를 하는 것일까? 이들은 주로 채용 전략 수립 및 실행, 우수 인재 Sourcing, 이들의 관리 및 검증 등의 업무를 수행한다. 우리 회사에 아주 적합한 역량을 가진 유능한 개발자가 있으면 적극적으로 채용에 나서기도 한다. 최근 스타트업

에서도 인하우스 리크루터를 많이 채용하는 추세이며 해당 분야의 헤드헌팅 업무 경험이 있는 사람을 선호하는 경향이 있다.

인하우스 리크루터를 활용하면 어떤 장점이 있을까?

첫째, 인하우스 리크루터의 적극성은 회사의 이미지 형성에 지대한 영향을 준다. 유능한 개발자들을 채용하기 위해 열정적이고 적극적으로 노력하는 이들의 모습을 외부인이 보면 우리 회사가 다른 회사와는 차별화된 점이 있다고 생각하기 쉽다. 또한, 회사의 이미지가 좋지 않다면 이를 전환하는 데에도 큰 기여를 할 수 있다.

둘째, 비교적 잘 알려지지 않은 스타트업은 리크루터를 통해 회사의 인지도를 높일 수 있다. 인하우스 리크루터의 역량 및 노력을 통해 무엇을 하는 회사인지, 어떤 업무를 하는지, 복지제도는 좋은지 등을 효과적으로 알릴 수 있다.

넘쳐나는 구직 정보

최근 변화를 살펴보면 후보자들은 더 똑똑해졌다. 예전에는 똑똑하지 않았다는 소리가 아니다. 더욱 스마트하게 변했다는 것이다. 원래는 회사가 지원자의 정보를 더욱 잘 아는 것이 일반적이었는데, 요즘에는 반대가 되었다. 왜 그런 현상이 발생할까?

우선 채용 정보를 얻을 수 있는 채널이 매우 다양해졌다. 기존에는 사람인 등의 채용 사이트에서 한정된 채용을 진행했다면 지금은 커뮤니티, 지인 추천 등 다양한 채널을 통해 채용 정보 접근이 가능하다. 또한, 페이스북이나 링크드인 같은 SNS를 통해 직·간접적으로 여러 채

용 관련 정보가 자연스럽게 노출된다. 잡플래닛과 같은 사이트에서 회사 연봉, 면접 후기 등도 상세하게 확인할 수 있다. 즉, 이직을 원하는 개발자들이 이러한 다양한 정보를 바탕으로 자신에게 맞는 적절한 회사를 골라서 갈 수 있는 상황임을 의미한다.

이러한 다양한 정보를 기반으로 다음의 두 가지를 판단하여 이직의 중요한 요소로 활용하기도 한다. 먼저 '새로움'에 대해서 소개한다. 후보자에게 그 회사의 포지션이 언제 열렸는지는 매우 중요한 포인트다. 만약 6개월 전에 올라온 채용 포지션에서 아직도 채용을 못 했다는 것은 부정적인 의미로 받아들여질 수밖에 없다. 반면, 하루 이틀 된 채용 공고는 이들이 새로움을 느끼기에 충분하다. 다음으로 '유일성'이다. 만약 개발자 본인에게 두 명 정도의 헤드헌터가 연락이 온다면 '너무 사람을 급하게 찾고 있구나', '시장에 마구 뿌리고 있구나', '나에게만 오는 특별한 기회가 아니구나'라는 느낌을 받을 수 있다. 개발자들은 다양한 채용 정보들을 기반으로 이 두 가지를 충분히 고려하여 본인이 회사를 스마트하게 판단할 수 있는 시대로 변화하였다. 마구잡이로 뿌려서 채용하던 시대는 점점 끝나가고 있다고 해도 과언이 아니다. 선택과 집중이 필요한 이유가 바로 여기에 있다.

그리고 카운터오퍼(Counter Offer)를 통해서 연봉 인상에 적극적인 태도를 보이는 경우도 많아졌다. 과거에는 여러 회사를 지원하고도 한 회사에만 지원했다고 선의의 거짓말을 하는 경우가 많았다. 하지만 지금은 달라졌다. 이제는 다른 회사와도 협상을 진행 중이며 연봉을 얼마 정도 맞춰주면 우리 회사에서 올 수 있다는 적극적인 태도를 보인

다. 이를 통한 연봉 인상 시도는 개발자 입장에서 성공할 확률이 매우 높기 때문이다.

비대면 채용의 확산

코로나19로 인해 화상 면접과 같은 비대면 채용이 빠르게 늘어나고 있다. 화상 면접은 기업뿐만 아니라 공무원 채용까지도 적용되고 있는 추세이다. 최근 경기도 안산시에서는 공무원 채용 시 화상 면접을 도입하기도 했다. 2020년 8월 인크루트에서 기업 인사담당자 224명을 대상으로 설문 조사를 진행하였는데, 2020년 하반기에 화상 면접 등 비대면 채용전형을 도입할 것이라고 응답한 비율이 67.3%였다. 이처럼 비대면 채용이 확대되면서 우리 기업에서도 이를 적극적으로 수용해야 하는 시기가 왔다.

물론 화상 면접은 기업뿐만 아니라 지원자도 부담을 느끼는 것이 사실이다. 지원자 입장에서는 이에 대한 정보가 많이 없으므로 ‘어떻게 준비해야 하는지’, ‘어느 장소에서 면접을 봐야 하는지’ 등 다양한 요소들에 대한 고민이 생길 수밖에 없다. 기업 입장에서는 면접을 원활하게 진행할 수 있는 플랫폼을 잘 준비해 놓아야 하고, 비대면 질문 방식을 통해서도 적절한 지원자를 효과적으로 걸러낼 수 있는 방법에 대한 고민이 필요하다.

이러한 어려움에도 불구하고 화상 면접 등의 비대면 채용은 더욱 확대될 전망이다. 최근에는 인공지능 면접으로 지원자를 1차 필터링하는 방법을 활용하는 기업도 많아졌다. 인공지능 면접의 신뢰도 등

에 의문을 제기하는 사람들도 있지만, 비대면 채용이 전체 채용 과정 중 일부로 자리매김하고 있는 것은 확실하다. 기업에서는 비대면 채용 방법에 대해 잘 알고 적극적으로 활용할 수 있는 준비가 필요한 시점이다.

개발자는 여전히 없다

부동산에는 매도자 우위 시장과 매수자 우위 시장이라는 말이 있다. 매도자 우위 시장은 사려는 사람은 많은데 팔 사람이 없는 것이고 매수자 우위 시장은 그 반대의 개념이다. 개발자 세계에도 위 공식은 그대로 적용될 수 있다. 기업과 개발자 둘을 각각 위 공식에 대입해보면 현시점은 누구 우위 시장일까? 두말할 거 없이 개발자 우위 시장이 형성되고 있다.

실제로 전체 소프트웨어 기업 중 약 20%의 포지션이 비어있는 상황이고 계속하여 개발자 채용을 원하고 있다. 일명 ‘네카라쿠배’라는 기업들도 개발자 채용 문제에서 자유롭지 못한 것이 현실이다. 하지만 이들은 막강한 복지와 연봉으로 무장한 기업들이다. 이들이 강력한 무기를 갖고 있다는 것을 절대 잊지 말아야 한다.

우리 주변에 개발자들이 많아 보이지만 막상 우리 회사에서 이들을 채용하려면 양적, 질적으로 부족하다. 우리 회사에 딱 맞는 개발자 찾기는 정말 하늘에서 별 따기라는 마음으로 채용에 임해야 한다. 개발자는 항상 부족하다는 사실을 명심하라. 개발자 우위 시장은 앞으로도 계속될 것이다.

Action Item

개발자 채용 시장에 대한 이해

- ✔ 개발자들의 이직은 조직의 크기와 관계없으며 이들의 문화이므로 이해해야 한다.
- ✔ 개발자가 없으면 스타트업도 없다는 생각으로 개발자를 대하라.
- ✔ 새로운 개발자 집단인 MZ세대 개발자들을 이해하고 공감하라.
- ✔ 채용은 한 번에 이루어지는 것이 아니고 꾸준한 준비가 필요하다.
- ✔ 채용을 위한 적극적인 구매와 이를 위한 준비가 필요하다.
- ✔ 인하우스 리크루터를 적극 활용하라.
- ✔ 넘쳐나는 구직 정보 속에서 기업이 좋은 이미지를 가질 수 있도록 잘 관리하라.
- ✔ 비대면 채용 등의 사회적 변화를 받아들여라.
- ✔ 개발자는 항상 부족하다는 생각으로 성의껏 채용을 진행하라.



PART

2.

개발자를 맞이할 준비는 되어있는가?

01 | 좋은 인재 채용을 위한 마케팅

02 | Selling Point를 잡아라



01

좋은 인재 채용을 위한 마케팅

채용 마케팅은 무엇일까?

당신이 살아가면서 마케팅이라는 말은 정말 많이 들어봤을 것이다. 그런데 채용 마케팅은 대체 무엇인가? 채용에 마케팅이 왜 필요한 것일까? 회사에서 월급을 주는데 왜 마케팅까지 해야 할까? 라는 의문이 든다면 이번 챕터를 반드시 정독하기를 바란다.

대기업도 하는 채용 마케팅

우리가 이름만 대면 아는 대기업은 어떤 방식으로 채용 마케팅을 진행하고 있을까? 대기업 공채 시즌이 되면 주요 대기업에서는 기다렸다는 듯이 학교별로 채용 설명회를 진행한다. 회사의 채용 일정과 인원, 방법 등 여러 정보를 제공하여 지원자들에게 우리 회사에 대해 구체적으로 설명한다. 그렇다면 대기업은 지원자가 적어서 이런 채용 설명회를 매번 진행하는 것일까? 사실 굳이 설명회를 진행하지 않아도 대

기업은 많은 구직자가 선호하는 직장이므로 채용에 큰 문제는 없다.

그렇다면 대기업들이 굳이 시간과 비용을 들여 채용 설명회를 하는 이유는 무엇일까? 가장 큰 이유는 ‘채용 설명회’라는 채널을 통해 우리 기업을 더욱 효과적으로 홍보할 수 있기 때문이다. 기업 홍보를 하는 수많은 방법이 있겠지만 ‘채용 설명회’는 타 홍보 방법에 비해 기업의 이미지를 보다 긍정적이고 효과적으로 알릴 수 있는 수단이다. 그리고 이를 통해 우수한 역량을 가진 학생이 지원할 확률을 높이고 이러한 지원자 중 필요한 인재를 채용하여 장기적으로 기업이 더욱 바람직한 방향으로 나아갈 수 있는 기반을 마련하는 것이다.

회사의 핵심 가치를 알려라

무조건 우리 회사를 마케팅한다면 유능한 인재를 채용할 수 있을까? 마케팅을 하기 전에 우리 회사에 대해서 생각해볼 필요가 있다. 우리 회사는 과연 매력적인가? 어떤 부분이 매력적인가? 이 질문에 바로 답변할 수 있다면 당신 회사는 구직자들이 매력을 느낄만한 요소가 충분히 많은 기업이다. 반면 매력적인 부분이 없거나 구직자들이 끌리는 무언가가 없다면 어떨까? 매력이 전혀 없는 회사를 아무리 마케팅한다고 해서 개발자들이 오고 싶은 회사가 될 수 있을까? 전혀 그렇지 않을 것이다. 먼저 우리 회사를 매력적인 회사로 만든 후에야만 효과적인 마케팅이 가능하다. 그러므로 기업은 매력적인 기업이 될 수 있도록 체질 관리를 잘해야 한다. 이를 이해하기 위해 필요한 것이 바로 EVP(Employee Value Proposition: 임직원 가치제안)이다.

EVP는 기업에서 제공하는 업무 환경, 동료, 보상 체계, 기업 문화 등을 총체적으로 의미하는 말이며 직장 평판의 총체라고 생각하면 쉽다. 또한, 내부 직원뿐만 아니라 외부에서 느끼게 되는 우리 회사에 대한 이미지라고 생각할 수도 있다. EVP는 왜 중요할까? 우선 바람직한 EVP를 가진 기업은 내부 직원들의 만족도가 높다. 이는 이들의 이직률을 현저하게 낮추는 요인이 된다. 채용 후보자의 95%는 입사 혹은 이직 시 회사의 평판을 상당히 중요하게 여긴다고 한다. 잘 관리된 EVP로 인하여 이들이 입사하고 싶은 회사로 만들 수 있다. 이는 우수 인재를 영입할 수 있다는 뜻이기도 하다.

그렇다면 바람직한 EVP 형성을 위해서는 어떻게 해야 할까? 이를 위한 방법을 소개한다.

첫째, 사전 조사가 필요하다. 경쟁사의 채용 전략 및 채용 브랜드에 대한 실태 조사를 진행하는 것이 좋다. 둘째, 현재 우리 회사 구성원에 대한 인터뷰를 진행한다. 이때 입사 전과 입사 후를 나눠 인터뷰하는 것이 좋다. 입사 전 질문 내용은 ‘채용 공고를 접한 채널’, ‘공고에서 끌렸던 문구’, ‘비공식 커뮤니티 등에서 알게 된 회사의 인상적인 정보’ 등을, 입사 후에는 ‘입사하여 알게 된 회사의 좋은 점’, ‘기존 채용 공고에서 아쉬웠던 점’ 등을 질문하는 것이 좋다. 셋째, EVP를 작성하고 이를 배포한다. 이때, 회사가 알리고 싶은 내용과 지원자가 알고 싶은 내용은 무엇인지 고민하여 이를 위주로 작성하면 좋다. 스토리텔링 형식으로 작성하는 것이 좋으며 솔직하고 구체적으로 작성한다. 이미지 혹은 카툰 형식으로 보기 편하게 작성하는 것도 좋으며 핵심 키워

드 위주로 구성해야 한다. 공고 전에는 외부 인력을 대상으로 어떤 문구와 형식이 더 매력적인지 테스트를 진행하여 선호도가 높은 쪽을 선택하는 것이 좋다. 또한, 이를 배포할 때에는 자사 블로그나 notion, brunch 등의 다양한 미디어를 활용하는 것이 좋다. 그리고 한 번 배포했다고 끝이 아니고 지속적인 수정과 보완 작업을 하는 것도 잊지 말아야 한다.

차별화된 채용 마케팅을 하라

이 글을 읽고 있는 대부분의 사람은 채용 문제에 대해서 많은 고민을 하고 있을 것이다. 또한, 우리 기업이 대기업 수준의 규모나 복지 등을 무기로 내세울 수 없을 가능성이 높다. 사실 채용 시장에서 중소/중견기업, 스타트업이 대기업보다 좋은 인재를 확보할 수 있는 확률은 많이 낮은 편이다. 그렇다면 우리 회사는 매번 대기업에 좋은 인재를 다 빼앗기기만 해야 할까?

실제로 주변을 보면 스타트업에 유능한 개발자나 인재들이 입사하여 일을 하는 경우가 생각보다 많다. 그렇다면 이들은 왜 스타트업에서 일을 하고 있는 것일까? 대기업에 갈 실력이 부족해서? 정답은 그렇지 않다. 이런 사람들의 대부분은 스타트업의 성장 가능성과 수평적 문화, 자율성 등의 장점을 보고 이직한 경우가 많다. 그렇다면 이들은 스타트업에 근무하기 전에 어떤 경로를 통해 기업의 정보를 듣고 왔을까? 단순히 지인을 통해서 정보를 얻었을까? 아니면 직접 구글링을 통해서? 아니다. 이들도 기업의 채용 마케팅에 이끌려 이 자리까지 온 것

이고 만약 해당 기업의 채용 마케팅이 없었다면 이 자리에 없었을 확률이 높다.

채용 마케팅이라는 것은 구체적으로 무엇일까? 채용 마케팅은 기업이라는 상품을 우수 인재들에게 널리 알려져서 기업의 가치와 이미지를 개선하고 결과적으로 기업이 우수 인재를 채용하는 데 긍정적인 영향을 주는 마케팅 기법으로 정의할 수 있다. 즉, 기업은 채용 마케팅을 통해 회사가 가진 가치와 비전, 기업 문화 등을 홍보나 채용 공고 등을 통해 자연스럽게 알리게 된다. 이를 통해 우리 기업이 상당히 매력적이고 입사하고 싶은 기업이라는 이미지를 각인시키는 것이다. 단순히 기업 정보, 문화 등을 소개하고 채용 공고를 내는 것도 채용 마케팅이라고 할 수 있다. 하지만 우리 기업은 대기업이 아니므로 이런 평범한 채용 마케팅으로는 경쟁 기업보다 우위를 선점할 수 없다. 차별화된 채용 마케팅 전략이 필요한 이유다. 이때 필요한 것이 바로 브랜딩이다. 우리 회사의 성장 가능성, 자유로운 기업 문화, 신나게 일할 수 있는 근무 환경 등 다양한 요소를 기반으로 브랜딩을 진행하는 것이다.

채용 브랜딩을 통해 유능한 인재를 사로잡자

우리 회사 채용을 브랜딩하라

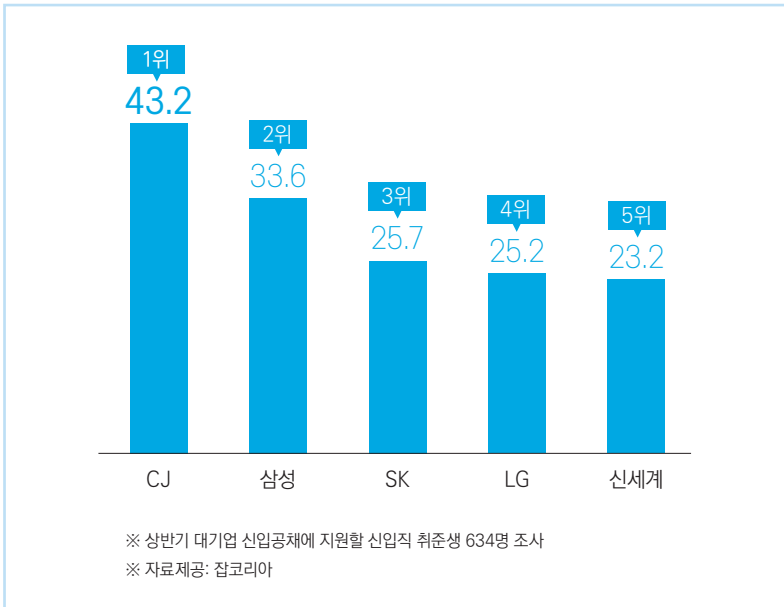
아이폰은 신제품이 출시되기만 하면 사람들은 이를 구매하기 위해 긴 줄을 서는 것을 마다하지 않는다. 여성들은 샵넬에서 신상품 가방

이 출시되거나 가격이 인상된다는 것을 알게 되면 길게 줄을 서서 그 제품을 구하기 위해 애를 쓴다. 이러한 현상은 왜 일어날까? 바로 브랜드의 힘이다. 해당 브랜드 제품을 구매하면 그 값어치만큼 구매한 사람의 가치도 높아질 것으로 생각하기 때문이다. 이렇게 사람들에게 브랜드는 매우 중요하며 이를 활용하면 채용 마케팅에서 우위를 선점할 수 있다.

앞서 언급한 기업은 기업의 이미지를 고급화하는 전략을 통해 좋은 제품을 높은 가격에 판매할 수 있게 된다. 그렇다면 우리 회사는 왜 브랜딩이 필요할까? 우리 회사가 판매하는 제품이 잘 팔리게 하는 목적도 있겠지만, 또 다른 목적이 있다. 바로 좋은 인재를 채용하는 목적을 이루기 위해서다. 여러분 주변에 구글에 근무하는 개발자가 있다고 해보자. 지인들은 구글이라는 세계 최고의 기업에서 최고의 기업 문화와 복지 등 다양한 혜택을 누릴 '개발자'를 부러워할 것이다. 그럼 구글이 이렇게 좋은 기업이라는 것은 누가 알려줘서 알게 된 것인가? 이는 다양한 매체 등을 통해서 구글이라는 기업의 근무 환경이 매우 좋고 최고의 복지제도를 제공한다는 사실이 꾸준히 노출된 결과라고 할 수 있다.

잡코리아의 조사에 따르면 대기업 신입 공채를 준비하는 취업준비생들이 가고 싶어 하는 기업 1위는 CJ(2018년)가 차지했다. CJ는 다양한 채널을 통해 기업 브랜딩을 매우 잘하는 회사로 알려져 있다. CJ는 기존의 딱딱한 채용 설명회 방식이 아닌 토크쇼, 화상 채팅과 같은 새로운 방식과 채널을 활용하여 기업 이미지 브랜딩에 힘을 썼다. 회사에서 일하는 선배와 솔직한 토크쇼 등을 통해 회사의 복지 등에 대한 다

양한 정보를 생생하게 전달했다. 2018년 상반기에는 ‘윤식당’이라는 TV 프로그램을 모티브로 한 ‘CJ의 맛있는 채용 이야기 JOB식당’을 유튜브를 통해 중계했다. 이에 대한 반응은 예상외로 뜨거웠고, 취업준비생뿐만 아니라 일반인들에게도 CJ에 대한 긍정적인 이미지와 기업 문화를 전할 수 있었다.



대기업 신입공채 취업목표 그들 1위 ‘CJ’

이렇게 기업의 이미지를 멋지게 만드는 브랜딩 작업은 우수한 인재 채용과 직결될 수밖에 없다. 근무 환경이 좋고 발전 가능성이 큰 기업이라는 브랜드를 가진 기업은 자연스레 지원자들이 선호하는 직장이

될 것이고 시간이 갈수록 더 유능한 인재들이 지원하는 선순환 체계가 마련될 수밖에 없다. 반대로 직원을 소모품 취급하며 야근이 잦은 회사라는 이미지로 기업 브랜딩이 된다면 결과는 어떨까? 이를 모르고 지원하는 직원들도 있겠지만 이러한 이미지를 아는 사람들은 절대 해당 기업의 채용에 지원하지 않을 것이다. 그렇다면 여러분의 회사는 어떤 이미지의 회사로 브랜딩하고 싶은가? 당연히 전자와 같은 긍정적인 이미지로 브랜딩하길 원할 것이며 이를 위해서는 다른 기업과 다른 차별화된 전략이 필요하다.

참신한 아이디어와 이색적인 마케팅 방법을 찾아라

수년 전부터 스타트업은 항상 개발자가 부족한 상황이 지속되었다. 초기 스타트업은 신규 서비스 개발을 위해 개발자가 필요하고 서비스 런칭이 된 스타트업은 서비스 확장을 위해 개발자가 더 필요한 상황이 지속되었다. 이렇게 스타트업에서 개발자는 절대 없어서는 안 되는 필수적인 존재다.

하지만 스타트업의 채용 시장은 그렇게 녹록지 않다. 앞서 언급한 것처럼 좋은 개발자는 스타트업의 생존을 위해서 반드시 필요하지만 반대로 유능한 개발자 입장에서는 스타트업에서 근무해야 할 이유가 많지 않다. 그래서 스타트업에서 일반적인 채용 공고를 내고 좋은 인재가 지원하기를 기다리면 생각보다 이력서가 잘 들어오지 않는 경험을 하게 된다. 왜 이런 현상이 발생할까? 냉정하게 말하면 스타트업과 같이 규모가 크지 않은 기업은 대기업에 비해 직업의 안정성이 매우 떨어진

다. 쉽게 말해 서비스가 잘못되면 바로 문을 닫을 수도 있는 기업인 셈이다. 개발자 입장에서 이런 위험 요소를 안고 이 세계에 몸담는다는 것은 쉽지 않은 결정이다. 또한, 회사 자체가 크지 않기 때문에 연봉이나 복지 부분에서도 대기업에 밀릴 수밖에 없다.

그렇다면 대기업이 비해 부족한 것이 많으니 우리 기업은 브랜딩을 포기해야 할까? 아니면 공격적인 채용 마케팅을 진행해야 할까? 스타트업이나 작은 규모의 기업은 우리 회사에 입사했을 때 큰 규모의 기업보다 더 좋은 점을 효과적으로 부각시켜야 한다. 그중 개발자 채용에 가장 중요한 요소로 작용하는 것이 바로 '성장 가능성'이며 이를 강조해 기업 이미지를 브랜딩할 필요가 있다.

대기업이 따라 할 수 없는 새로운 방법으로 채용 공고를 진행해보는 것은 어떨까? 이를 위해 스타트업만의 색깔로 색다른 채용을 진행한 사례를 소개한다. '강백호 같은 인재를 찾습니다'라는 독특한 채용 공고를 통해 성공적으로 유능한 개발자들을 채용한 스타트업 '슬로그업'의 사례는 스타트업 업계에서 나름 유명하다. 아래는 그 당시 슬로그업이 냈던 채용 공고 중 일부다.

일반적인 채용 공고를 냈을 때는 지원자가 한 명도 없다가 위와 같은 채용 공고를 낸 후 반응이 매우 폭발적이었다고 한다. 화려한 이력을 가진 지원자들의 지원서 수십 장이 접수되었고 실제로 30명 이상의 지원자를 대상으로 면접까지 진행할 수 있었다고 한다. 아직 서비스를 런칭해본 적도 없는 신생 스타트업이 이렇게 성공적으로 채용을 한 것은 이례적인 일이었다. 이는 정성스럽게 준비한 채용 공고와 기업이 추



**복산 같은
스타트업에서
강백호 같은
개발자를
찾습니다.**

누구에게나 팀이 필요합니다.
마음 맞는 팀이 필요합니다.
만약 복산을 만나지 않았다면
강백호의 인생은
한날 철모르는 양아치로
끝났을 겁니다.

정대만도 그냥 동네 강배였을지 모릅니다.
실력 좋은 서태웅도 마찬가지입니다.
실력이야 나무랄 데 없겠지만,
모난 성격을 받아준 복산이 아니라
다른 팀이었다면 실력을 온전히 발휘할 수
있었을까요? '스타트업계의 복산'을 같이
만들어갈 분을 찾습니다.

부족한 실력을 근성으로 이겨내는
강백호

목목하게 말은 바 임무를 완수하는
채치수

실력으로 끌리는 건 절대 용납 못 하는
서태웅

불꽃 같은 열정으로 밤새워 나오는
정대만

단신 원디립을 노력과 스피드로 극복한
송대섭

이 다섯 사람 중 한 명의 모습이
자신과 닮았다면 연락주세요.
두 달 발려 환영합니다.

구하는 가치에 대한 브랜딩이 얼마나 중요한 것인지 깨닫게 해준다.

채용 마케팅 잘하기

그렇다면 우리 회사를 더욱 돋보이게 할 채용 마케팅을 어떻게 하는
게 가장 효율적일까? 구체적인 순서와 함께 상세한 내용을 소개한다.

● 우리 회사 탐색하기

가장 먼저 우리 회사의 현황을 분석해야 한다. 내부의 직원을 대상으로 인터뷰 등을 진행하여 우리 회사가 어떤 기업인지 파악하는 일이 우선이다. 우리 회사는 직원들이 가장 잘 알고 있으므로 현황 분석에 상당한 도움이 된다. 그리고 회사의 대외적인 이미지를 채용 포털 사이트, SNS, 뉴스 등 다양한 매체를 통해 파악하는 것이 좋다. 이때 소셜 빅데이터 분석을 통해 대중들이 느끼는 감정 등을 분석하는 것도 도움이 된다. 만약 회사의 규모가 크지 않다면, 회사가 서비스를 진행할 대표 상품의 주요 키워드를 기반으로 연관어 분석 등을 진행하면 효과적인 파악이 가능하다.

● 브랜딩 이미지 결정하기

탐색하기 단계에서 대내외적으로 수집한 내용을 바탕으로 회사를 대표할 수 있는 브랜딩 이미지를 결정한다. 예를 들면 ‘개인과 회사가 함께 성장하는 회사’, ‘누구나 자유로운 의견을 낼 수 있는 회사’, ‘퇴근할 때는 인사하지 않는 회사’ 등 우리 회사가 추구하는 가치를 자연스럽게 나타낼 수 있는 메시지를 정리한다.

● 브랜딩 콘텐츠 제작

앞서 개발한 브랜딩 메시지를 활용하여 실제 콘텐츠를 개발하는 단계이다. 예를 들어 ‘누구나 자유로운 의견을 낼 수 있는 회사’라는 브랜딩 메시지를 통해 콘텐츠를 개발한다고 해보자. 콘텐츠는 크게 사업

내용 및 회사를 소개할 수 있는 콘텐츠와 회사 내 직원 등을 대상으로 실제 경험 등을 소개하는 콘텐츠로 나누면 효과적이다. 전자의 콘텐츠는 우리 회사가 어떤 제품을 판매하는 회사인지, 어떤 가치로 운영되는지, 협업은 어떻게 진행되는지 등 기업 정보 및 생활 전반에 대한 내용을 소개한다. 후자의 콘텐츠는 직원들을 대상으로 한 인터뷰를 통해 회사 생활, 연차 사용, 복지, 연봉 등 공식적으로 언급하기 힘든 경험담을 다양하게 소개한다. 이런 구성을 통해 공식적으로 회사를 소개하는 콘텐츠와 형식이 비교적 자유로운 인터뷰 형식의 콘텐츠가 시너지 효과를 발휘하여 우리 회사 이미지 브랜딩을 위한 훌륭한 콘텐츠가 마련된다.

● 브랜딩 콘텐츠 알리기

콘텐츠가 완성되었으면 이제 많은 사람들에게 알리는 단계이다. 이를 위해 다양한 채널을 통해 홍보를 진행한다. 대표적인 홍보 채널로는 SNS, 구직 플랫폼(잡플래닛, 사람인), 해당 직군 커뮤니티(개발자 커뮤니티, 개발자 네이버 카페) 등을 활용할 수 있다. 해당 커뮤니티는 개발자들이 적극적으로 활동하는 곳이라 더욱 효과적으로 노출이 가능하다.

● 피드백하기

앞서 다양한 채널을 통해 콘텐츠를 노출한 후 이를 보고 내부 직원과 외부에서 느껴지는 감정 등을 피드백 받은 후 이를 다음 홍보와 브

랜딩 작업 시 적극적으로 참고하여 반영할 수 있도록 한다. 이는 더 나은 브랜딩 작업을 위한 필수적인 작업으로 이 단계를 소홀히 하면 결과적으로 기업의 긍정적인 이미지 및 브랜딩 형성에 부정적인 영향을 미칠 확률이 매우 높다.

● 기업 평판 관리하기

배달의 민족, 요기요 등과 같은 배달 플랫폼에 입점한 업체들이 가장 중요하게 생각하는 것은 무엇일까? 바로 고객의 평점과 리뷰 관리이다. 한두 고객의 불만족스러운 리뷰는 소비자에게 좋지 않은 이미지를 심어주고 이는 매출로 직결된다. 이를 위해 업체들은 다양한 서비스 혜택을 제시하면서 높은 평점과 좋은 리뷰를 부탁하기도 한다.

채용 시장에도 이러한 트렌드가 확산되고 있다. 지원자들은 누구나 잡플래닛, 크레딧잡과 같은 사이트에서 기업의 평판과 평점 등을 조회할 수 있다. 지금 바로 위 사이트에 들어가서 우리 기업이 어떤 평판을 받고 있는지 확인해라. 이름을 알고 있는 주변 기업들의 평판도 조회하면 상당히 흥미롭다. 여기에는 생각보다 회사 내부를 적나라하게 볼 수 있는 내용이 많이 적혀있다. ‘근무 환경이 좋고, 출퇴근이 자유로워 좋다’, ‘수평적인 문화가 좋으며, 개인 발전도 충분히 가능하다’ 등의 좋은 이야기가 나올 수 있고 ‘CEO 마인드가 별로라서 빨리 이직하고 싶다’, ‘면접 볼 때 지원자를 하대하여 문을 박차고 나왔다’ 등의 좋지 않은 이야기도 가득하다.

당신의 회사는 어느 쪽인가? 긍정적인 평판이 많으면 좋겠지만 반대로 부정적인 평판이 많다면 우리 회사에 악영향을 미칠 확률이 높다. 이제 기업도 배달 업체처럼 평점 관리를 적극적으로 해야 하는 시대가 왔다.

그렇다면 평점이 높은 기업과 낮은 기업은 어떤 차이가 날까? 평점이 높은 기업들은 대부분 채용 프로세스가 상당히 빠르고 채용 진행 시 매끄러운 커뮤니케이션 관계를 유지한다. 면접에 대한 안내가 상세하며 면접 시에도 지원자를 존중하며 따뜻한 분위기로 진행한다. 반면, 평점이 낮은 기업들은 정해진 채용 프로세스의 시간을 지키지 않으며 지원자에 대한 대우를 전혀 하지 않고 하대하기도 한다. 또한, 채용 프로세스도 생각보다 엉망으로 진행되며 잦은 말 바꿈 등으로 신뢰를 잃는 경우가 많다.

수년 전만 해도 지원자들은 회사 내부의 조직 문화, 분위기, 복지 등을 알기가 쉽지 않았다. 하지만 요즘에는 대부분의 지원자가 회사 평판을 보고 지원한다. 이런 이유로 평판 관련 사이트를 잘 관리해야 한다. 만약 좋지 않은 평판이 있으면 내부 직원들의 좋은 경험담을 올리는 것도 좋다. 물론 거짓을 올리는 것은 역효과가 날 수 있다. 배달 업체가 고객 한 명을 소중하게 여기는 것처럼 지원자를 소중하게 여기고 대우해주는 노력이 필요하다.

소프트웨어 산업의 이해

우리 회사는 어떤 유형의 기업일까? 어떤 서비스를 하는 기업일까? 게임 소프트웨어를 만드는 기업일 수도 있고 응용 소프트웨어를 만드는 기업일 수도 있다. 기업은 유형별로 다른 마케팅 기법을 활용할 필요가 있다. 다른 기업에서 성공한 마케팅 방법이라고 똑같이 따라 하다가는 큰코다치기 마련이다. 효율적인 마케팅 전략 수립을 위해서는 최소한 우리 기업이 소프트웨어 산업 내 어느 위치에 있고 어떤 서비스를 하는지 정도는 알아야 한다. 따라서 이번 장에서는 이에 대한 상세한 내용을 알아보려고 한다. 아래 자료는 소프트웨어정책연구소에서 매년 발간하는 소프트웨어산업실태조사 내용을 발췌 및 요약한 자료이다.

소프트웨어 산업 분류체계

해당 조사에서는 소프트웨어 산업을 패키지 소프트웨어, IT 서비스, 게임 소프트웨어, 인터넷 소프트웨어(정보서비스) 총 4개 영역으로 분류하였으며 상세 내용은 아래와 같다.

● 패키지 소프트웨어

패키지 소프트웨어는 기성 제품으로써 완성된 형태로 제공되는 소프트웨어를 의미하며 세부적으로는 시스템 소프트웨어와 응용 소프트웨어로 구분된다.

- **시스템 소프트웨어:** 컴퓨터 시스템을 효율적으로 운영해주는 소프트웨어로 운영체제 소프트웨어, 보안 소프트웨어, 스토리지 소프트웨어, IT 운영관리 소프트웨어, 데이터 분석 및 관리 소프트웨어, 미들웨어 소프트웨어, 애플리케이션 개발 및 테스트 소프트웨어 등이 포함된다.
- **응용 소프트웨어:** 특정 응용 분야의 정보처리 활동을 수행하거나 사용자에게 특정 업무 기능을 제공하는 소프트웨어로 개인용 소프트웨어, 산업 범용 소프트웨어(콘텐츠 관련 소프트웨어, ERP/ERM 소프트웨어, CRM 소프트웨어, SCM/SRM 소프트웨어), 산업 특화 소프트웨어(금융 소프트웨어, 제조 소프트웨어, 에너지 소프트웨어)가 포함된다.

● IT 서비스

IT 서비스는 최적의 정보기술을 활용하여 조직·업무·사업의 부가 가치를 제고하고 정보기술 기반 기존 산업과의 융·복합화를 통해 새로운 서비스를 창출하는 영역을 의미한다. 세부적으로는 IT 컨설팅, 정보시스템 개발 및 통합, 정보시스템 관리 및 운영 서비스가 포함된다.

- **IT 컨설팅 및 정보시스템 개발 서비스:** 기업의 인적·물적 자원 등 관련 자료를 수집·분석하고 최근 정보기술 활용 자원과 현재의 정

보를 고려하여 정보시스템을 개발·구축·통합하거나 이에 대한 기술적 조언·자문을 수행하는 서비스다.

- **IT 시스템 관리 및 지원 서비스:** IT 컨설팅과 IT 시스템 설계 및 개발을 통해 개발·구축·통합된 정보시스템에 대해 운영 및 관리, 유지보수, 교육·훈련 등을 제공하는 서비스다. IT 시스템 관리와 BPO(Business Process Outsourcing) 서비스, IT 지원 서비스가 포함된다.

● 게임 소프트웨어

컴퓨터·비디오·게임용 소프트웨어 및 데이터를 의미한다.

- 게임 소프트웨어에는 유선 온라인 게임 소프트웨어, 모바일 게임 소프트웨어, PC 게임 소프트웨어, 비디오 게임 소프트웨어, 아케이드 게임 소프트웨어가 포함되며 장르는 롤플레이, 보드, 전략, 퍼즐, 액션, 스포츠, 시뮬레이션, 카드 게임 등이 있다.
- **유선 온라인 게임 소프트웨어:** 인터넷을 통해 멀티플레이가 가능하도록 고안된 멀티미디어형 게임이다.
- **모바일 게임 소프트웨어:** 스마트폰, 태블릿PC 등을 통해 플레이가 가능하도록 고안된 게임이다.

● 인터넷 소프트웨어(정보서비스)

인터넷상에서 정보 검색 서비스, 커뮤니티 서비스, 데이터 저장 서비스 등을 제공하는 포털과 영상·음향·출판·게임·교육·뉴스 등의 정보

매개 서비스업, 호스팅 및 관련 서비스업, 데이터베이스 및 온라인정보 제공업을 총칭한다.

소프트웨어 유지 관리

소프트웨어의 성능 개선, 장애 방지 등 생명주기연장으로 시스템 재 구축 비용을 줄이고 운영 효율화와 비용 절감을 돕는 활동이다. 소프트웨어 유지 관리의 주요 활동은 패치 서비스, 업데이트, 일상지원, 긴급 장애처리, 예방/예측 지원, 고객맞춤지원, 업그레이드와 같은 활동으로 상세한 내용은 아래와 같다.

- **패치 서비스:** 새로운 기술의 적용이나 운영체제의 변화 등으로 발생하는 불일치를 조정하는 서비스이다.
- **업데이트:** 기존 소프트웨어 제품의 기능을 보완하기 위하여 추가 되는 서비스이다.
- **일상지원:** 전화 및 e-mail, 온라인 지원 등을 통한 질의응답 서비스이다.
- **긴급 장애처리:** 사용자가 장애처리 및 정비 서비스를 요청하는 경우 문제를 해결하는 서비스이다.
- **예방/예측지원:** 시스템 장애를 사전에 예방하기 위해 정기적으로 지원하는 서비스이다.
- **고객 맞춤지원:** 주변 환경에 적합하도록 맞추는 커스터마이제이션, 마이그레이션 서비스이다.

- **업그레이드:** 기존 소프트웨어 제품을 향상시키기 위하여 새로운 버전으로 교체하는 서비스 기술이다.

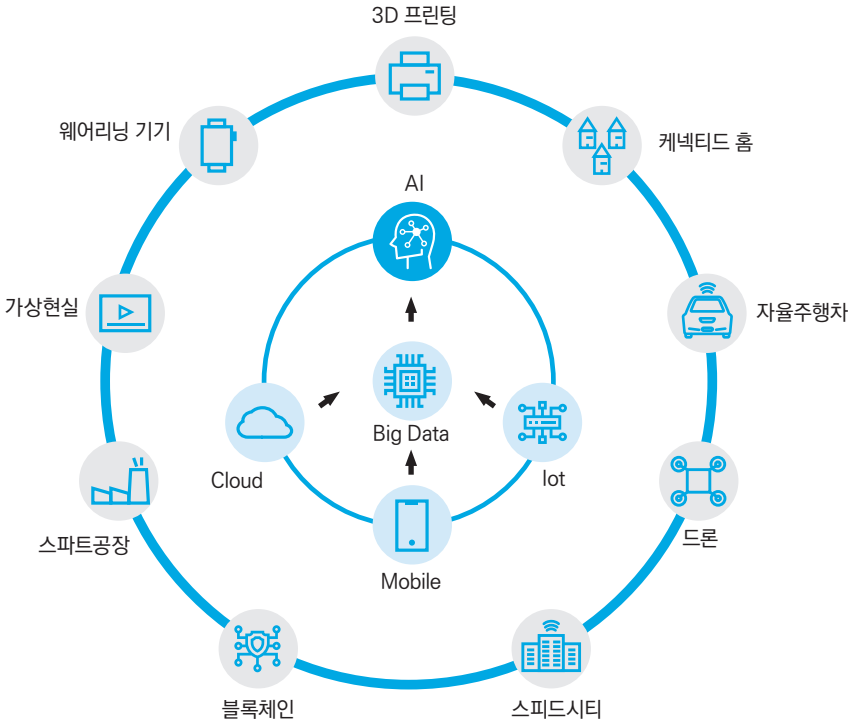
기술개발 환경

● 하드웨어 플랫폼

- **서버/스토리지/PC/노트북:** 제공되는 제품이나 서비스가 서버/스토리지/PC/노트북을 기반으로 소프트웨어나 서비스가 제공되는 모델이다.
- **무선통신기기:** 무선통신기기(스마트폰, 웨어러블 디바이스)를 통해 소프트웨어나 서비스가 제공되는 모델이다.
- **전용 단말기:** 제공되는 제품이나 서비스가 특정 목적의 전용 단말기나 시스템을 기반으로 소프트웨어나 서비스가 제공되는 모델이다.

● 소프트웨어 플랫폼

- **Unix:** 동시에 다중 사용자/다중 태스크의 실행을 지원할 수 있는 것을 특징으로 하는 대화형의 운영체제이며 텍스트 조작 툴, 문서 처리, 전자 메일 외에 취급이 쉬운 파일 시스템을 갖추었다.
- **iOS:** 애플이 개발 및 제공하는 임베디드 운영체제로, 아이폰, 아이패드, 아이팟터치, 아이패드, 애플 TV 등에 탑재되어 구동되는 운영체제이다.



- **Linux:** 대형 기종에서만 작동하던 유닉스를 개인용 컴퓨터(PC)에서도 작동할 수 있게 만든 운영체제로 인터넷을 통해 프로그램 소스 코드를 무료로 공개하여 사용자는 원하는 대로 특정 기능을 추가할 수 있을 뿐만 아니라 어느 플랫폼에도 포팅이 가능하다.
- **RTOS:** 디스크를 이용한 다중 프로그래밍 운영체제로 사용자가 특유의 실시간 소프트웨어를 구성하는 데 필요한 기본 기능을 제공하는 운영체제이다.

- **공개소프트웨어(OSS, Open Source Software)**

소프트웨어의 내용을 프로그래밍 언어로 나타낸 소스코드(Source Code)를 공개하여 누구나 개량·재배포할 수 있는 소프트웨어를 의미한다.

신소프트웨어

신소프트웨어는 인공지능, 블록체인, 클라우드 VR, IoT 등 4차 산업시대에 필요한 중요한 기술이며 매우 급격하게 변하는 분야이므로 그 범위를 구분하고 한정하기 어렵다. 따라서 아직까지 명확한 정의를 내리기 어려운 상황이다. 본 가이드북에서는 각 유형이 어떤 기술인지를 간략하게 소개하는 정도로 마무리하고 더욱 상세한 내용은 다양한 자료를 통해 확인하길 바란다.

- **클라우드 서비스**

인터넷 기반의 컴퓨팅 서비스를 의미하며 인터넷에 연결된 대용량 서버·스토리지를 기반으로 정보 분석 및 처리, 저장, 관리, 유통 등의 서비스를 제공하는 것을 의미한다. 컴퓨팅 자원을 서비스하는 IaaS(Infrastructure as a Service), 개발과 관련된 표준화된 플랫폼을 제공하는 PaaS(Platform as a Service), 소프트웨어 및 관련 데이터는 서버·스토리지에 호스팅되고, 사용자가 클라이언트 단말을 통해 접속하는 형태의 SaaS(Software as a Service) 등이 포함된다.

● 빅데이터 분석

디지털 환경에서 생성되는 방대한 규모의 정형·비정형 데이터 자체를 의미하기도 하며 혹은 빅데이터를 수집, 저장, 처리, 분석할 수 있는 플랫폼이나 소프트웨어를 통해 빅데이터를 분석하고 이를 통해 예측하는 서비스를 총칭한다. 빅데이터 분석 소프트웨어 툴과 애플리케이션, 그리고 이를 활용하는 다양한 서비스 등이 포함된다.

● IoT

유·무선 네트워크를 통해 센서·단말 등이 연결되어 정보를 공유하는 환경을 의미하며 IoT를 구현하기 위한 소프트웨어 플랫폼 및 IoT 환경을 통해 유통되는 정보, 콘텐츠 등이 포함된다. IoT 분석 및 응용 소프트웨어와 IoT 플랫폼, IoT 하드웨어, IoT 보안 등이 포함된다.

● 인공지능(AI)

인공지능이란 컴퓨터에서 인간과 같이 사고·학습하고 판단하는 논리적인 방식을 사용하는 인간의 지능을 본뜬 고급 컴퓨터프로그램을 의미한다. 인공지능 플랫폼, 애플리케이션, IT 서비스, 비즈니스 서비스, 인공지능 하드웨어, 칩 등이 포함된다.

● XR(VR/AR/MR)

VR(Virtual Reality, 가상현실)은 자신(객체)과 배경·환경 모두 현실이 아닌 가상의 이미지를 사용하는 기술을 의미한다. AR(Augmented

Reality, 증강현실)은 현실의 이미지나 배경에 3차원 가상 이미지를 겹쳐서 하나의 영상으로 보여주는 기술이다. MR(Mixed Reality, 혼합현실)은 현실세계와 가상세계가 혼합된 상태를 말하며 AR(증강현실)과 가상현실에 현실 정보를 추가하는 AV(Augmented Virtuality, 증강가상현실)의 의미를 포함한다. 최근에는 이러한 기술들을 통칭하여 XR(Extended Reality, 확장현실)이라고 한다. XR 플랫폼, 콘텐츠, 컨설팅, 시스템 통합, 뷰어 등이 포함된다.

● 융합 신서비스

독립적으로 제공되는 서비스의 주요 기술 등을 결합하여 새로운 형태의 가치를 제공해주는 서비스이다. 스마트팩토리, 스마트헬스케어, 스마트에너지, 스마트팜, 인터넷은행/핀테크, 자율주행, 3D 프린팅 등이 포함된다.

● 블록체인

온라인 금융 거래 정보를 블록으로 연결하여 P2P(Peer to Peer) 네트워크 분산 환경에서 중앙 관리 서버가 아닌 참여자(Peer)들의 개인 디지털 장비에 분산-저장시켜 공동으로 관리하는 방식을 의미한다. 블록체인 플랫폼, 애플리케이션, 비즈니스 서비스, 서버, 보안 등이 포함된다.

02

Selling Point를 잡아라

개발자, 당신은 누구인가?

우리 회사에서 개발자 채용이 필요하다면 우선적으로 그들에 대한 이해가 필요하다. 지금까지 우리 회사에서 봐 왔던 직원들과 비슷한 특성을 지녔다고 단정하고 이들을 대하면 조만간 회사를 떠날지도 모른다. ‘지피지기면 백전백승’이라고 했다. 그래서 이번 장에서는 개발자들이 가진 특성과 그들만의 문화를 구체적으로 소개한다.

개인이 발전할 기회를 좋아한다

- 성장 가능성을 중시한다

개발자는 다른 전문직과 비슷한 특징을 가졌다고 보면 쉽게 이해할 수 있다. 목표가 주어지면 스스로 할 일을 찾아내고 다양한 자료를 통해 학습을 하며, 목표를 달성해 나가는 것을 좋아한다. 그러므로 개발자 대부분은 회사 내의 영향력 있고 혁신적인 프로젝트에 참여하고 싶

어 한다. 단순하게 참여하는 것이 아닌 적극적인 프로젝트 참여를 통해 자신이 얼마나 성장할 수 있는지가 중요한 포인트이다.

- **새로운 경험을 좋아한다**

이들은 새로운 경험을 매우 중요하게 생각한다. 우리 회사에서 개발하는 서비스가 비교적 쉽게 누구나 개발할 수 있는 정도의 난이도라면 개발자들이 좋아할까? 개발자들은 본인 커리어나 경험에 도움이 되지 않는 프로젝트를 좋아하지 않는다. 이러한 개발 경험이 아무리 많더라도 본인의 발전에 전혀 도움이 되지 않기 때문이다. 지금까지 경험해보지 못했던 새로운 유형의 경험을 제공할 수 있다면 이들은 적극적으로 해당 프로젝트에 참여하여 본인의 역량을 마음껏 발휘할 것이다.

- **꾸준한 학습을 추구한다**

개발자 대부분은 새로운 것을 학습하는 것에 대한 두려움이 별로 없다. 최신 기술이 나오면 그것을 공부하기 위하여 꾸준한 노력을 아끼지 않는다. 만약 이들이 회사 업무에 필요하다며 서적이거나 관련 자료를 구매하고 싶다고 한다면 절대 아끼지 말고 적극적으로 지원하라. 또한, 본인의 전문 분야와 관련된 연수나 세미나, 콘퍼런스 등에 참석하고 싶다고 하면 무조건 갈 수 있도록 비용과 시간을 지원한다. 여기에 투자한 비용은 나중에 몇 배가 되어 우리 회사의 중요한 자산으로 돌아올 것이다. 또한, 고급 수준의 개발자로 올라갈수록 한 가지 기술에

대한 전문성보다는 전체 IT 분야의 흐름을 파악하는 것이 중요하다. 이때도 다양한 자료와 여러 분야의 강의 등을 활용하여 많은 지식을 습득할 수 있도록 도움을 주어야 한다. 다시 말하면, 다양한 최신 기술 등을 끊임없이 학습할 수 있는 환경 조성에 도움을 주고 나머지는 개발자 본인에게 맡기면 된다는 얘기다.

개발을 위한 환경이 잘 구축되어 있어야 한다

● 업무 효율성 향상을 위한 툴을 잘 갖춰야 한다

일반적으로 업무용 커뮤니케이션 툴은 이메일을 많이 활용해왔다. 이메일은 급하지 않은 업무에 대한 커뮤니케이션을 진행하는 툴이다. 회신을 받는 데에도 시간이 많이 걸린다.

물론 이러한 이메일 기반의 소통도 나쁘지 않고 아직도 많이 활용하는 방식이다. 하지만 개발자들은 새로운 협업 툴을 활용하여 업무의 효율성이 높아진다면 활용법을 배우는 것을 주저하지 않는다. 최근 기업들은 다양한 업무용 메신저, 공유 드라이브, 협업 툴(Slack, Notion, MS Teams, 잔디, Google Workspace, Github, Jira, Confluence)을 활용하고 있다. 이러한 툴을 활용하면 업무에 대한 빠른 회신이 가능하여 일 처리가 수월해진다. 실시간 공동 작업이 가능해 굳이 모이지 않아도 각자 원격 환경에서 업무를 수행할 수 있다는 장점도 있다. 소규모 회사에서는 큰 비용 부담 없이 사용할 수 있고 일정 인원 혹은 무료 용량 이상을 활용하기 위해서는 추가로 비용을 부담해야 한다. 하지만 회사 내 빠르고 효율적인 커뮤니케이션을 위해서는 필수적으로

도입해야 하는 툴이다. 앞서 언급한 툴 중 각 회사의 상황에 맞는 툴을 선택하여 활용하는 편이 좋다.

이전에 경험하지 못했던 것이라 할지라도 이 툴의 장점과 활용법 등에 대해 간략하게 설명해주면 개발자들은 빠르게 습득하고 적극적으로 사용한다. 반대로 말하면, 업무의 효율성을 높이는 도구들을 활용하지 못하면 회사 업무의 효율성이 매우 떨어진다고 생각을 할 것이다.

내일 회의 자료를 만드는데 직원 대부분이 협업 툴 활용에 능숙하지 않아 계속 모여서 회의를 한다고 가정해보자. 물론 모여서 회의가 필요한 사안도 있겠지만 이보다 더 중요한 것은 일의 효율성이다. 개발자들은 일의 능률을 매우 중요시하기 때문에 이러한 상황을 효율적으로 해결할 수 있는 협업 툴을 가장 먼저 찾을 것이다.

하지만 회사에서 이런 툴의 비용 등을 절약하려 하거나 팀원 대부분이 활용에 능숙하지 못하다면 이들의 업무 만족도는 크게 저하될 것이다. 개발자들이 새로운 업무용 툴을 제안한다면 회사에서는 이를 활용하는 방안을 적극적으로 검토해야 한다. 이들이 제안한 새로운 툴을 활용함으로써 회사의 업무가 더욱 효율적으로 개선될 확률이 매우 높기 때문이다.

● 개발하는 장비가 좋아야 한다

당연한 이야기지만 개발에 필요한 컴퓨터, 모니터, 마우스, 키보드 등은 최고로 편하고 좋은 제품으로 지원하라. 이들 장비는 창의력을 바탕으로 새로운 서비스를 만들어내는 도구이다. 주방장은 좋은 칼,

골프 선수는 좋은 골프채, 카레이서는 좋은 차가 준비돼야 한다. 개발자에게 좋은 개발 장비가 있어야 최고의 개발 성과가 나올 수 있다. 또한 조금이라도 불편한 부분이 발생하지 않도록 책상과 의자, 스탠드 등의 사무용 가구나 주변 기기에도 많은 배려를 해야 한다. 만약 업무에 새로운 사무용 기기나 장비가 필요하다면 이 또한 아끼지 말고 적극적으로 지원하라. 이들은 이를 활용하여 몇 배의 성과를 낼 수 있을 것이다.

- 최고의 업무 공간을 제공하라

개발자들은 업무 공간에 무척 민감하다. 개발자들에게 있어 업무 공간은 고도의 집중을 요구하고 이러한 몰입을 통해 최고의 산출물을 낼 수 있는 장소다. 개발자들의 이러한 특성을 알기에 많은 기업들은 개발에 몰두할 수 있는 최고의 환경을 제공한다. 이를 위해 1인이 활용할 수 있는 사무실을 제공하기도 하고 그렇지 못하다면 타인의 방해 를 최소화할 수 있는 업무 환경을 제공한다. 또 다른 유형으로는 집에서 자유롭게 개발할 수 있도록 배려해줄 수 있으며 집 근처의 공유 오피스 등을 제공하기도 한다. 만약 우리 회사에서 이러한 환경을 제공 하지 못한다면 당장 개발자들이 근무할 수 있는 환경으로 만들 필요가 있다.

좋은 동료들이 있어야 한다.

● 탁월한 동료는 좋은 성과로 이어진다

넷플릭스 최고인사책임자인 패티 매크드(Patty McCord)는 “최고의 직장은 복지가 좋거나 급여가 많은 곳이 아니라 탁월한 동료들과 함께 일할 수 있는 곳”이라고 말했다. 이 말은 개발자 세계에는 더욱 확실하게 적용된다. 개발자들은 본인이 최고인 것보다는 더 유능한 개발자가 있는 회사를 선호한다. 이름만 들으면 알만한 개발자가 근무하는 곳이라면 서로 가고 싶어 한다. 그 이유는 무엇일까? 바로 동료들 통해 다양한 것들을 배우며 성장할 수 있기 때문이다. 또한 이런 동료와 일하게 되면 능률도 향상되고, 흥미로운 결과물도 낼 수 있는 여러 장점이 있다. 만약 A라는 개발자는 코딩 속도가 매우 빠르고, B라는 개발자는 디버깅을 잘하며, C라는 개발자는 전체 시스템 통합 작업을 잘한다고 해보자. 모두 각자 분야에서 최고의 역량을 보이는 개발자들이지만 특정 분야 이외의 것들은 잘하지 못한다는 단점이 있다. 하지만 이들이 상호 협력하여 장점을 극대화하고 단점을 보완한다면 그 효율은 이루 말할 수 없을 정도로 높아질 것이다. 이것이 바로 같이 일하는 동료의 중요성이며 결국 회사의 생산성 향상에 많은 도움을 주게 된다. 때문에 회사에서는 인재 관리를 통해 좋은 동료와 일할 확률을 높이는 것이 중요하다.

개발자가 전혀 없는 회사에서 처음 개발자를 뽑는다면 이들은 절대 가고 싶어 하지 않는다. 이런 회사에 가면 본인이 결정한 대로 모든 것이 흘러갈 수는 있지만 자기 발전에는 큰 도움이 되지 않는다고 생각

한다. 또한, 같이 개발 업무를 담당할 동료들이 전혀 없으므로 혼자 난관을 헤쳐나가야 할 수 있다. 따라서 이런 회사는 개발자들이 선호하지 않는다. 개발자들은 함께 즐겁게 일하며 서로에게 배울 수 있는 곳에서 근무하는 것을 바란다. 이런 환경은 하루 이틀 사이에 만들 수 있는 것이 아니므로 꾸준한 노력을 통해 일궈내야 하는 결과물이다.

높은 자율성은 최고의 책임감을 만든다

● 근무 장소와 시간에 자율성을 부여하라

개발자들은 과제가 정해지면 스스로 목표를 잡아서 개발하는 경향이 짙다. 따라서 개발자가 가지고 있는 이러한 경향에 맞춰 근무 환경도 바뀌어야 한다. 최근 많은 스타트업에서 개발자들에게 근무 환경에 대한 자유를 주고 있다. 필요하다면 언제든지 재택 근무를 활용할 수 있게 하며 출퇴근 시간을 자유롭게 운영하기도 한다. 또한, 연차 휴가 등이 필요하면 언제든지 눈치 보지 않고 자유롭게 쓸 수 있게 한다. 단, 본인과 회사가 설정한 목표에 도달하기 위해서 주어진 업무를 효율적으로 수행해야 하는 책임이 있다. 이러한 책임을 다하지 않고 위의 권리만을 누리라는 것은 절대 아니다. 만약 우리 회사에 이러한 제도가 없다면 다양한 스타트업의 사례를 살펴보고 적극적으로 반영할 필요가 있다.

● 본인에게 많은 결정권을 주어라

회사 생활을 하다 보면 연차 활용, 법인 카드 사용, 재택 근무 사용

등 상위 결정권자의 결재가 필요한 경우가 많다. 물론 결재가 정말 필요한 경우도 있겠지만 결재 과정 때문에 번거롭고 눈치가 보여 이런 제도를 잘 활용하지 못하는 경우도 많다. 최근 스타트업 등에서 활용하는 제도 중 '본인 전결 제도'가 있다. 결재가 필요한 상황이면 단순 통보 형식으로 결재를 올리고 본인이 결재를 하여 최종 책임을 스스로 가지는 방식이다. 이는 복잡한 과정의 결재를 쉽게 하자는 취지도 있지만 개발자 본인이 스스로 결정할 수 있는 주체라는 것을 인식시켜 업무에 대한 책임감과 소속감을 높이는 방법이기도 하다.

회사의 개발 문화를 중시한다

개발 문화를 어떻게 정의할 수 있을까? 사실 개발 문화라는 것은 추상적이며 우리가 원한다고 해서 한 번에 만들어질 수 있는 것도 아니다. 개발자들은 지원하는 회사의 개발 문화에 대해 궁금해하며 면접 시에 주로 이러한 내용을 물어본다. 일할 회사의 개발 문화를 알아야 본인이 회사에서 적응할 수 있을지를 판단할 수 있기 때문이다. 개발 문화는 개발하는 코드에 대한 꾸준한 리뷰, 에지일 프로세스를 활용한 개발, 비슷한 성향을 가진 동료와의 협업 등 구성원들이 함께 공유하고 실행해 나가야 할 목표와도 같다. 좀 더 부드럽게 표현하면 개발을 하면서 같이 지켜나가야 할 공감대 형성이라고 할 수 있다. 즉, 이런 목표와 공감대 형성을 통해 개발 문화가 만들어진다.

언급한 것들을 무조건 적용한다고 해서 개발 문화가 바로 형성되는 것은 아니다. 단순히 개발만 열심히 하는 것이 아니라 궁극적으로 우

리 회사 서비스의 품질(코드의 품질)을 향상하려는 공감대가 형성되는 것이 개발 문화의 시작이라고 볼 수 있다. 이러한 것들을 개발자 모두가 이해하고 서로 지켜나가고자 할 때, 비로서 개발 문화로 자연스럽게 정착된다. 회사는 이러한 개발 문화가 잘 정착할 수 있도록 애써야 한다. 만약 우리 회사에 이러한 문화가 제대로 정착되지 않았다면 이를 위한 부단한 노력이 필요하다.

편안한 근무를 위한 최고의 복지가 필요하다

● 법인 카드를 아끼지 말라

개발자들은 생각할 시간이 많이 필요하며 이를 통해 창의적인 결과물을 산출한다. 이들에게 시간은 매우 소중하며 이러한 시간을 아낄 수 있게끔 도움을 주어야 한다. 이를 위해 개발자들이 회사 업무와 연계하여 필요한 물품이 있을 때는 적극 지원하도록 하자.

어떤 회사는 필요한 물품을 회사 공용 아이디 장바구니에 담아놓으면 다음 날 도착할 수 있도록 바로 구매한다고 한다. 이는 개발자들이 물품 구매를 위한 비용을 아낄 수 있도록 하기도 하지만 이를 구매하기 위한 고민과 쏟는 시간을 절약하는 데 많은 도움을 준다. 출출할 때 먹을 수 있는 간식과 음료 등을 갖추는 것도 잊지 말아야 한다. 회사에서 식사해야 한다면 법인 카드로 맛있는 식사를 할 수 있도록 해주는 것이 좋다. 그런 자유를 준다고 해서 이들이 법인 카드를 흥청망청 쓰는 일은 거의 없다. 자유의 범위가 늘어나는 대신 이들은 책임감 있게 이를 활용할 것이므로 크게 걱정하지 않아도 된다. 최고의 효율

을 내기 위한 공간에서 배가 고파서 혹은 식대가 부담스러워서 아무 것도 못 하면 안 되지 않겠는가? '볍카'는 이럴 때 쓰라고 있는 것이다. 아낌없이 주는 회사가 되자.

● 개발자의 가족도 소중하다

개발자들이 장기근속하며 회사를 발전시킬 수 있는 가장 큰 요인 중 하나는 이들의 가족이다. 반대로 말해서 이들의 가족에게 회사의 복지가 영향을 미칠 수 있다면 개발자들은 더욱 만족할 것이며 회사의 성과 향상에 기여를 할 확률이 높아지게 된다.

구체적으로 어린 자녀가 있는 경우 사내 어린이집이나 공동 어린이 집 등에 아이를 맡길 수 있게 해주는 것이 좋다. 개발자 본인의 건강과 가족 건강 모두를 챙길 수 있도록 건강검진, 체력 단련 비용 등을 지원하는 것도 좋다. 가족 여행을 갈 수 있도록 콘도 및 호텔 이용권을 주기적으로 제공하는 것도 가족 복지 혜택 중 하나다. 일주일 중 하루를 가족의 날로 정하여 조기 퇴근 혹은 영화 쿠폰 등을 제공하는 것도 좋은 방법이다. 개발자의 가족을 사랑하는 것이 바로 개발자를 아껴 주는 일이고 이는 장기적으로 회사의 발전과 연계될 것이다.

● 충분한 휴식을 할 수 있는 공간이 필요하다

개발 업무는 고도의 집중을 통한 작업이 이루어지므로 피로감 또한 그에 비례한다. 이런 이유로 충분한 휴식을 취할 수 있는 환경이 필요하다. 국내외 여러 기업에서 개발자들이 휴식을 취할 수 있는 환경

을 마련하고 있다. 비디오 게임을 하며 머리를 식힐 수 있게 하거나 만화책을 마음껏 볼 수 있는 공간을 제공한다. 숙면을 할 수 있는 공간을 제공하기도 한다. 또한, 마사지를 받으며 휴식을 취할 수 있는 환경을 제공하는 곳도 있고 당구/탁구 등 실내 스포츠를 하며 스트레스를 풀 수 있는 공간을 제공하는 회사도 있다. 이러한 환경은 결국에 개발자들의 업무 집중도 향상과 생산성 향상에 기인하게 되므로 이런 공간 마련에 아낌없이 투자해야 한다.

높은 연봉은 좋은 개발자 채용으로 이어질 확률이 높다

제일 중요한 이야기지만 제일 마지막에서 언급하는 이유는 개발자마다 높은 가치를 두는 곳이 다르기 때문이다. 실력이 좋은 개발자는 어디를 가도 당연히 연봉을 많이 받는다. 우리 회사에서 실력 있는 경력 개발자를 뽑기 위해서는 시장에서 해당 경력자들의 연봉 수준보다는 높게 지급할 준비를 해야 한다. 중고차 시장에는 이런 말이 있다. '싸고 좋은 차는 없다'. 개발자 세계도 마찬가지다. 연봉이 낮으면 좋은 개발자를 구할 확률은 그렇게 높지 않다. 만약 회사가 앞에서 언급한 조건들을 모두 다 갖추고 연봉만 조금 부족하다면 이야기가 다르겠지만 그런 기업이 개발자에게 줄 연봉이 부족하다는 것은 말이 되지 않는다. 좋은 개발자를 구하려면 그에 맞는 연봉을 충분히 주어야 이들의 마음을 움직일 수 있고, 회사에 오더라도 이직할 확률이 낮아진다. 넷플릭스는 동종 업계 최고의 연봉으로 A급 직원들을 뽑아서 운영한다고 한다. 또한, 주기적으로 시장 조사를 하여 해당 개발자의 연봉 시

세를 파악하여 더 높은 연봉을 주는 곳이 있으면 그 기업보다 더 많은 연봉으로 협상한다고 한다. 이는 최고의 직원을 뽑아서 유지하고 관리하는 데도 많은 노력을 기울인다는 얘기다. 우리 회사에서 뽑을 개발자의 등급을 정했다면 최소한 그에 맞는 평균 이상의 연봉을 준비하고 영입할 준비를 해야 한다. 그렇지 않으면 절대 우리가 원하는 수준의 개발자를 뽑을 수 없다.

개발자들에게 알려라

우리 회사가 채용을 진행하고 있다는 사실을 많은 개발자들에게 알리기 위해서는 어떤 방법을 활용하는 것이 좋을까? 개발자 직군은 개발하는 서비스, 개발 환경, 활용 언어 등에 따라서 다른 특색이 있다. 채용 전략은 이에 따라서 꾸준히 변해야 한다. 단순히 '채용 공고 사이트에 올리면 개발자들이 이것을 보고 우리 회사에 지원하겠지'라는 안일한 생각을 가지면 안 된다. 그만큼 시대가 변했고 개발자들도 바뀌었으며 채용 채널도 다양해졌다. 우리 회사의 채용 공고가 최대한 노출되는 방법들을 골라서 활용해야 한다. 하나의 채널만을 고집하면 절대 안 된다.

IT 리크루팅 및 인재 관리 회사인 테크시스템스의 리서치 디렉터 제이슨 헤이먼은 “잘못된 방법을 사용하거나 인재를 찾을 때 활용하는 채널이 충분하지 않은 회사들이 많다. 유지하고자 하는 기술과 인재

의 종류, 이들을 찾을 수 있는 장소를 생각해야 한다. 경력이 20년인 보안 분석가와 헬프 데스크 지원 인력을 찾는 전략이 같을 수는 없다. 반드시 달라야 한다”고 강조했다.

그렇다면 채용 채널은 어떤 것들이 있을까? 일반적으로 많이 활용하는 채용 채널을 소개하겠다.

● 소셜 미디어

소셜 미디어는 우리가 가장 많이 접하는 채널 중 하나다. 트위터, 페이스북, 링크드인 등이 이러한 채널에 속한다. 소셜 미디어에 채용 광고를 하고 링크드인에서 적합한 인재를 적극적으로 찾는 노력도 해야 한다. 개발자들은 본인의 커리어를 외부에 보이고 싶어 하므로 이러한 소셜 미디어를 많이 활용한다. 특히 링크드인 등의 소셜 미디어는 우리 회사에 필요한 인재를 효과적으로 검색하고 연락을 할 수 있는 수단으로 활용할 수 있다.

● 구인/구직(공채) 포털

가장 전통적인 방식의 공고 방식이다. 예전에는 신문 등을 통해서 구인 광고를 냈지만 이제는 구인/구직 포털을 통해서 원하는 인재를 찾는다. 개발자뿐만 아니라 대개의 채용 공고는 이러한 포털에 올라온다고 생각하면 된다. 우리가 일반적으로 많이 알고 있는 사람인, 인크루트 등이 이러한 채널에 속한다. 많은 사람들이 사용하는 만큼 빼놓을 수 없는 채널 중 하나이다.

● 세미나/콘퍼런스/박람회

세미나/콘퍼런스 등은 최근 코로나19 사태로 인해 많이 위축되었다. 이로 인해 온라인 콘퍼런스 등으로 대체되어 운영되고 있으나 이 또한 충분한 홍보 효과를 가져올 수 있다. 기업 체험관, 홍보관 등을 온/오프라인으로 운영하여 우리 회사를 대중에게 잘 알릴 수 있다. 이는 자연스럽게 회사의 문화나 분위기를 예비 채용 후보자들에게 전달할 좋은 기회다. 또한, 오프라인 행사 진행 시 현장에서 다양한 분야의 사람들을 만나볼 기회가 될 수 있다.

● 사내 채용

인력이 필요한 자리가 있다면 외부에 공고를 내기 전에 회사 내부에 채용할 사람이 있나 확인할 필요가 있다. 사내에 해당 포지션에 맞는 사람은 없는지, 적절한 교육을 통해 이러한 개발 업무를 수행할 수 있는 사람은 없는지 등을 먼저 고려해야 한다. 등잔 밑이 어둡다고 했다. 멀리서 찾지 말고 우선 안에서 찾아보면 적합한 인재가 있을 수 있다. 외부에서 개발자를 채용할 경우 어느 정도의 리스크를 갖게 되지만, 사내 개발자들은 이미 회사 문화에 적응했을뿐더러 인성 검증도 되어 있기 때문에 큰 장점이 될 수 있다.

● 직원 지인 추천

내부 직원이 외부의 유능한 인재를 추천하는 방법도 있다. 1차로 검증된 인재를 우선하여 채용할 수 있어 많은 회사에서 활용하는 방법

이다. 내부 직원 추천으로 외부 인재가 채용되면 두 직원 모두에게 보너스 등의 혜택을 지급하는 기업도 많아지고 있다. 아무것도 모르는 인력을 채용하는 것보다는 조금이나마 검증이 된 사람을 더욱 선호하는 것이다. 회사 입장에서는 잘못된 인력 채용만큼 고통스러운 일은 없기 때문이다.

● 커뮤니티 활용

개발자들은 새로운 분야에 대한 개발을 진행하면서 수많은 오류를 접하게 된다. 이에 관해 물어볼 사람이 없으면 혼자 구글링 등 다양한 방법으로 해결한다. 만약 오류를 접했을 때 같이 공유하고 해결할 수 있는 곳이 있으면 어떨까? 그래서 만들어진 것이 개발자 커뮤니티다. 해외는 '스택 오버플로'라는 사이트가 대표적이고 국내에서는 오픈소스 커뮤니티, SNS 중심의 커뮤니티, 포털서비스의 카페를 중심으로 한 커뮤니티들이 있다. 많은 개발자들이 가입해 활발히 활동하므로 이곳이야말로 금광이라고 부를 수 있다. 개발자들 대부분이 모여 있으므로 이곳을 집중적으로 공략하면 좋다. 회사 대표 중에는 개발자 커뮤니티 활동을 하는 경우도 많다. 커뮤니티 활동을 통해 왕성한 활동을 하는 사람을 관찰할 수 있고 이들의 성향, 실력 등을 파악할 수 있으며, 이를 기반으로 우리 회사에 채용할 수도 있기 때문이다.

그렇다면 개발자 커뮤니티를 처음 접한 사람이 바로 왕성한 활동을 할 수 있을까? 이를 위한 정답은 먼저 솔직해지는 것이다. 개발자들은 어느 직군의 사람들보다 순수하다고 생각한다. 만약 어떤 사람이 “나

는 30년간 농사만 짓고 있는 사람입니다. 스마트팜을 위해서 개발자가 너무 필요한 상황입니다. 어떻게 하면 될까요?”라고 글을 올리면 많은 개발자들이 이를 위한 조언을 해줄 것이다. 솔직함이 많은 사람들의 공감을 이끌 것이고 이러한 소통을 통해 커뮤니티에서 활동량을 늘리는 것이 좋다.

Developer Relations(이하 DevRel)

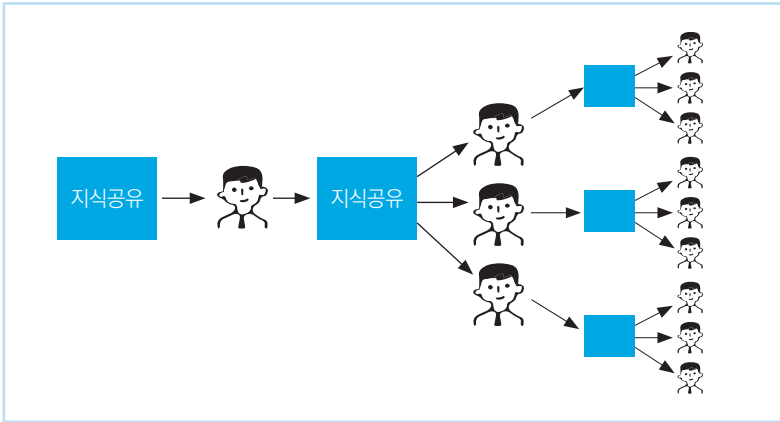
DevRel은 이런 도움을 준다

국내에서 DevRel 직군을 운영하고 있는 회사는 손에 꼽는다. 그렇다면 DevRel이란 무엇일까? DevRel은 ‘Developer Relations’의 줄임말로 개발자와 기업 간의 소통을 도와주는 역할을 한다. 즉, 개발자의 입장을 대변하고 개발자가 필요한 것들을 적시에 제공해주는 역할을 한다.

DevRel의 역할은 연예 기획사의 매니저라고 표현할 수 있다. 내부 개발자들이 외부에서 인정받을 수 있도록 중간자의 역할을 하는 셈이다. 예를 들어 외부의 유명한 콘퍼런스, 세미나, 해커톤 등에 우리 회사의 개발자들이 참석할 수 있게 하여 이들뿐만 아니라 회사의 명성도 높이는 역할을 한다. 만약 정부에서 주최하는 대규모 소프트웨어 개발자 행사에 우리 회사의 핵심 개발자가 기조 강연을 한다고 해보자. 해당 강연을 통해 개발자 본인뿐만 아니라 회사의 명성까지 높이는 계

기가 될 것이다. 이러한 역할을 바로 DevRel에서 하는 것이다.

DevRel은 왜 이러한 역할을 할까? 주된 이유는 좋은 개발자를 뽑을 수 있는 기반이 되기 때문이다. 내부 개발자들은 DevRel의 다양한 활동을 통해 성장할 수 있고 이는 회사의 발전과 연계된다. 외부 개발자들이 봤을 때에는 '나도 저 회사에 가면 저렇게 성장할 수 있구나'라는 생각이 들 것이다. 이는 유능한 개발자 채용, 회사의 평판 향상 등에 많은 도움을 줄 수 있다.



DevRel은 다음과 같이 운영하라

DevRel은 어떻게 운영해야 할까? 초반에는 내부의 리소스와 인맥 등을 적극적으로 활용할 필요가 있다. 처음부터 외부로 드러나는 활동을 할 필요는 없다. 회사의 인지도가 높지 않을 때는 외부의 인지도를 빌리는 방법이 있다. 이 분야에서 유명한 개발자들이 활동하는 커

뮤니티(파이썬 코리아, 케라스 코리아, 코리아 스타트업 포럼)에서 활동하면서
친분을 쌓는 것도 좋은 방법이다. 이러한 친분은 우리 기업의 인지도
향상에 많은 도움을 줄 수 있고 추후 홍보와 채용 등을 진행할 때 도
움을 받을 수도 있다.

해커톤을 활용하는 것도 좋은 방법이다. 해커톤은 회사에서 직접
주관할 수도 있고 멘토로 참여할 수도 있다. 회사에서 주관할 경우 다
양한 상급과 입사 시 유리한 점 등을 강조한 홍보가 가능하다. 입상자
및 수상작에 대한 홍보도 가능하므로 일거양득이다. 게다가 입상자의
실력이 이미 검증되었으므로 이들을 대상으로 빠른 채용 진행도 가능
하다. 실제로 회사에서 개최한 해커톤에서 우수한 성적을 거두고 채
용된 사례가 많다(몇 년 전 우아한 형제는 해커톤에 참여하기만 하면 1차 서
류 전형을 통과시켜주었다). 다음은 개발자들이 해커톤에 멘토로 참여하
는 경우다. 이때 우수한 개발자들이 적극적으로 참여할 수 있도록 한
다. 이들이 멘토링을 진행하면 해커톤 참여자에게 '이렇게 유능한 개
발자랑 일할 수 있겠구나', '이 회사는 비전이 있구나' 등의 긍정적인
이미지를 줄 수 있다. 기업 입장에서는 '참가자의 실력이 검증되었구
나', '이 정도 유능하면 우리 회사에서 바로 쓸 수 있는 인재구나' 등 지
원자에 대한 긍정적인 이미지를 가질 수 있다. 즉, 기업과 개발자 둘 다
win-win 할 수 있는 전략이다.

다음으로는 다양한 홍보 채널을 활용해야 한다. 우리 회사의 서비
스를 써볼 기회를 주고 여기에 참여하는 개발자들을 확보하는 것도
하나의 방법이다. 또한, 아주 사소한 내용이라도 블로그나 유튜브에

올리는 것도 좋은 방법이다. 만약 물류회사라면 ‘딤러닝을 활용해 우리 회사의 물류를 빠르게 하는 방법’ 등 호기심을 가지게 할만한 내용을 꾸준히 올린다. 이런 방법은 검색엔진에서 노출될 확률을 높이고 자연스럽게 회사 이미지 향상과 연계된다. 실제로 쏘카에서는 기술 블로그를 운영하고 있는데 효과가 매우 좋다고 한다. 기술 블로그를 통해 완성도 높은 성과, 회사가 만들어내는 최신 기술, 사내 프로젝트 진행 과정 등을 소개한다. 이는 회사의 인지도 향상뿐만 아니라 좋은 개발자 채용과도 연계된다. 다만, 자랑거리 등의 좋은 내용만 올리지 말고 개선이 필요한 부분도 함께 올리면 좋다. 이런 문제를 해결하기 위해서 개발자들이 우리 회사의 문을 두드릴지도 모른다.

다음은 비용 문제에 대해 생각해보자. 회사는 곧바로 성과로 이어지지 않는 부문에 대한 투자는 달가워하지 않는 편이다. 비용도 적지 않게 들 수 있어 꺼릴 수도 있다. 하지만 채용을 위해서는 DevRel 운영 비용 이상을 소모할 수 있다. 어차피 채용을 위해서 크든 작든 비용 투자를 해야 한다. 인재 확보를 위한 투자를 해야 한다면 DevRel을 효과적으로 운영하는 것이 더 나은 선택일지 모른다. 초기 비용이 조금 많이 들더라도 DevRel을 만들고 운영하는 데에 초점을 맞춰야 한다.

- ✔ 채용 마케팅은 대기업도 사활을 거는 중요한 홍보 수단이다.
- ✔ 개발자들에게 회사의 핵심 가치를 알리는 것이 중요하다.
- ✔ 다른 기업과 차별화된 채용 마케팅을 준비하고 실천하라.
- ✔ 우리 회사 채용을 매력적으로 브랜딩하는 것이 중요하다.
- ✔ 참신한 아이디어와 이색적인 마케팅 방법을 찾아라.
- ✔ 우리 회사를 돋보이게 하는 채용 마케팅을 체계적으로 진행하라.
- ✔ 개발자들은 개인이 발전할 수 있는 기회를 좋아한다.
- ✔ 개발을 위한 협업 톨, 장비 등 환경 구축에 많은 노력을 해야 한다.
- ✔ 개발자들은 보고 배울 수 있는 좋은 동료들이 있는 환경을 선호한다.
- ✔ 개발자들에게 높은 자율성을 주면 최고의 책임감으로 돌아온다.
- ✔ 개발자들은 회사의 개발 문화를 중시하므로 좋은 개발 문화 형성을 위한 많은 노력이 필요하다.
- ✔ 개발자들의 편안한 근무를 위한 최고의 복지제도를 마련하는 것이 좋다.
- ✔ 높은 연봉은 좋은 개발자 채용으로 이어질 확률이 높다.
- ✔ 보다 많은 개발자에게 우리 회사 채용을 알리기 위해서 다양한 채용 채널을 적극적으로 활용해야 한다.
- ✔ 개발자와 기업 간 소통하는 역할을 하는 DevRel은 회사와 개발자 모두에게 큰 도움을 준다.
- ✔ DevRel은 초기 운영비용 마련, 꾸준한 인맥 관리 및 홍보 등을 통해 효율적으로 운영하는 것이 좋다.



3.

어떻게 뽑을 것인가?

01 | 채용 준비하기

02 | 채용 실행하기



01

채용 준비하기

지금부터는 우리 회사에 필요한 개발자를 직접 채용하는 과정에 대해 알아보자. 일반적으로 개발자 채용의 시작은 조직 내부의 요구로부터 시작하기 마련이다. 새로운 사업을 준비 혹은 추진하기 위한 인력이 필요할 수도 있고, 특정 부서의 업무량이 증가하여 이를 보완해줄 인력이 필요할 수도 있다. 또한, 부서 내 기존 개발자가 다른 회사로 이직 혹은 퇴직할 경우 빈자리를 메워줄 인력이 필요하기도 하다. 채용 목적에 따라 우리가 뽑아야 하는 개발자의 성격은 달라지기 마련이다. 명확한 목적 없이 이루어지는 채용은 실패할 확률이 높다. 그러므로 우리는 가장 먼저 ‘왜 개발자를 뽑아야 하는가?’라는 질문에 대답할 수 있어야 한다.

채용 목적을 명확히 하라



채용 자체가 목적이 되어서는 안 된다

우선 우리 회사에 개발자가 왜 필요한지에 대한 구체적인 고민이 필요하다. 우리가 개발자를 채용하여 얻고자 하는 것은 무엇인가? 이 질문에 대한 대답이 필요하다.

단순하게 개발자 채용을 통해 디지털 전환을 해보려고, 혹은 시대에 뒤처지지 않기 위해서라고밖에 대답할 수 없다면 개발자 채용이 당장에 필요하지 않을 수도 있다. 단순히 개발자가 필요하다는 생각으로 채용을 하면 실패할 확률이 높다. 개발자 채용을 통해 어떤 목적을 달성할 것인지 구체화하고 이에 맞게 채용하는 것이 무엇보다도 중요하다. 하지만 기업에서는 '채용' 자체가 목적이 되는 경우가 흔히 발생한다. '공채 시기이기 때문에 직원을 뽑는다', '신규 사업에 필요할 것

같은 인력을 미리 확보해야 한다'는 목적을 갖고 채용을 진행하는 경우가 많다. 이럴 경우 심중팔구 우리 회사에 그다지 필요하지 않은 인력이 채용될 가능성이 크다. 우리가 목적하는 바와 다른 역량을 가진 개발자가 입사한다면 이들이 회사를 만족하며 다닐 수 있을까? 본인이 목적하는 바와 회사가 추구하는 가치가 다를 경우 다시 다른 회사로 이직을 결심하고 떠날 가능성이 매우 높다.

‘우리가 개발자를 채용하는 목적은 무엇인가?’라는 질문에 구체적인 대답을 내놓지 못하다면 지금은 개발자를 채용할 시기가 아니다. 반대로 명확한 답변이 가능하다면 바로 채용을 진행할 것을 추천한다.

채용을 통한 이익이 발생해야 한다

다음은 회사 비즈니스에 IT 기술 도입을 통해 얼마나 이익이 발생하는지에 대한 고려가 필요하다. 해당 개발자 채용을 통해 회사가 얼마나 이윤을 얻을 수 있는지 파악해야 한다. 기업의 입장에서는 채용을 통한 이윤이 발생하지 않는다면 굳이 진행할 필요가 없다.

우수한 개발자를 채용했다고 곧바로 기업 성장 및 이윤 창출과 연계되지 않을 수 있다. 단기적으로 봤을 때는 당연히 그럴 수 있다. 하지만 장기적인 관점에서는 우수한 개발자 채용이 회사의 이익에 많은 도움을 줄 것이다. 따라서 눈앞에 보이는 이익만을 추구하지 말고 해당 채용을 통해 얻어질 수 있는 장기적인 이익까지 고려하여 준비해야 한다.

목적이 없으면 채용 후 후회한다

기업의 채용담당자나 기업 대표 중 60%는 개발자의 채용 후 후회한 적이 있다고 한다. 원하는 스펙의 개발자를 찾아 여러 번의 면접을 거쳐서 채용했는데도 후회하는 일은 발생한다. 그만큼 회사의 목적에 맞는 개발자 채용은 쉽지 않다.

왜 이러한 현상이 발생하는 것일까? 가장 큰 원인은 기업의 실무와 개발자가 잘하는 분야가 일치하지 않기 때문이다. 아무리 코딩과 프로젝트 경험이 많은 개발자일지라도 기업의 업무에 부합하지 않는다면 우수한 성과를 내기는 어렵다. 이러한 결과는 명확한 목적을 설정하지 않고 채용을 진행하거나 스펙 위주로 채용을 진행했기 때문에 나타난 부작용 중 하나다.

우리 회사에서 개발자 채용을 통해 무엇을 할 것인지 등의 명확한 기준을 정한 후 채용을 진행해야 한다는 얘기다. 그래야 기업과 개발자 모두가 만족스러운 채용이 될 수 있다.

뽑을 수 있다면 한 번에 최소 두 명을 뽑는 것이 좋다

우리 회사에서 개발자를 채용할 때 몇 명을 뽑는 것이 좋을까? 필요한 사람이 한 명이라면 한 명만 뽑을 수도 있다. 스스로 일을 찾아서 할 수 있는 역량을 가진 시니어급 개발자는 한 명만 채용해도 큰 문제가 발생하지 않는다. 하지만 주니어급 개발자 한 명만 채용할 경우 문제가 발생할 확률이 높다. 주니어 개발자는 배워야 할 것이 많은 개발자다. 우리 회사에서 필요한 서비스를 개발하기 위해 공부만 하다가

퇴사할 확률이 높다. 혼자서 하는 개발은 힘들고 외로운 길이다. 게다가 비 IT 기업에서 대표와 다른 직원들을 이해시켜가며 성장하기란 절대 쉽지 않다.

개발자 채용시 한번 채용에 최소 2명 이상을 뽑으면 여러 가지 장점이 있다. 이렇게 2명을 채용하면 처음에는 성과가 70~80% 정도 밖에 나오지 않을 것이다. 하지만 바로 앞에 닥치는 문제만 바라볼 게 아니다. 이 두 개발자는 서로 의지할 수 있는, 친한 동료가 될 수 있다. 결과적으로 서로 협업하여 모두 성장할 수 있다. 둘 중에 한 명이 퇴사해도 남은 한 명이 일단 급한 업무는 맡아서 진행할 수 있으므로 업무 리스크를 크게 줄일 수 있는 장점도 있다.

사실 스타트업이나 중소기업에서 두 명을 한 번에 채용하는 게 쉬운 일은 아니다. 요즘엔 제대로 된 개발자 한 명만 찾아도 감사한 일이다. 하지만 대표가 꾸준하게 인맥 관리를 하고 다양한 노력을 한다면 어렵지 않게 두 명 이상의 유능한 개발자들을 채용할 수 있다. 이는 대표의 의지와 노력에 달려있다고 해도 과언이 아니다.

회사의 규모 및 상황에 따라 목적을 달리 설정해야 한다

개발자 관련 채용 공고를 보면 기업의 규모도 다양하고 업무도 천차만별이다. 기업의 규모 및 상황에 따라서 채용의 목적을 달리 설정할 필요가 있다.

- 초기 스타트업은 제품을 빠르게 만들 사람이 필요하다

초기 단계의 스타트업이 원하는 개발자는 제품을 빠르게 만드는 역량을 가진 사람이다. 많은 시간을 들여 완성도가 높은 제품을 만드는 것보다 시장에서 검증이 가능한 수준의 제품을 빠르게 만드는 것이 더욱 중요하다. 만약 초기 단계의 스타트업에서 완성도를 중시하는 개발자를 채용한다면 어떻게 될까? 기업 입장에서는 빠른 시일 내에 제품 출시가 어려워질 수 있다. 개발자 입장에서는 품질이 낮은 결과물을 출시하는 것에 대해 불만을 가질 수 있다. 그러므로 기업 규모가 크지 않은 스타트업은 높은 완성도보다는 빠른 제품/서비스 개발이 가능한 개발자를 채용해야 효율적이다. 버그가 조금 발생하더라도 런칭이 가능한 수준의 제품/서비스를 신속하게 만드는 역량을 가진 개발자를 뽑는 것이 유리하다.

- 서비스 고도화 단계 기업은 완성도 높은 제품을 만들 사람이 필요하다

서비스 고도화 단계 기업은 시간이 조금 걸리더라도 완성도 높은 서비스를 개발하는 일이 무엇보다 중요하다. 빨리 제품을 만드는 개발자보다는 버그 없이 완성도가 높은 서비스를 개발할 수 있는 개발자를 뽑는 것이 유리하다. 만약 이 유형의 기업에서 초기 단계 스타트업에 적합한 개발자를 채용한다면 어떨까? 해당 유형의 개발자도 일을 잘 해낼 수 있는 역량이 있겠지만 완성도가 높지 않을 수 있으며 여러 버그가 발생할 수 있다. 이로 인해 고급 경력 개발자가 추가로 투입되어

야 하는 상황이 발생할 수 있다. 따라서 이 유형의 기업에서는 높은 완성도로 제품/서비스를 개발할 수 있는 역량을 가진 개발자를 채용하는 편이 바람직하다. 기업의 유형에 따라 목적하는 바가 명확하게 설정이 되어야만 적합한 개발자를 채용할 수 있는 확률이 높아진다.

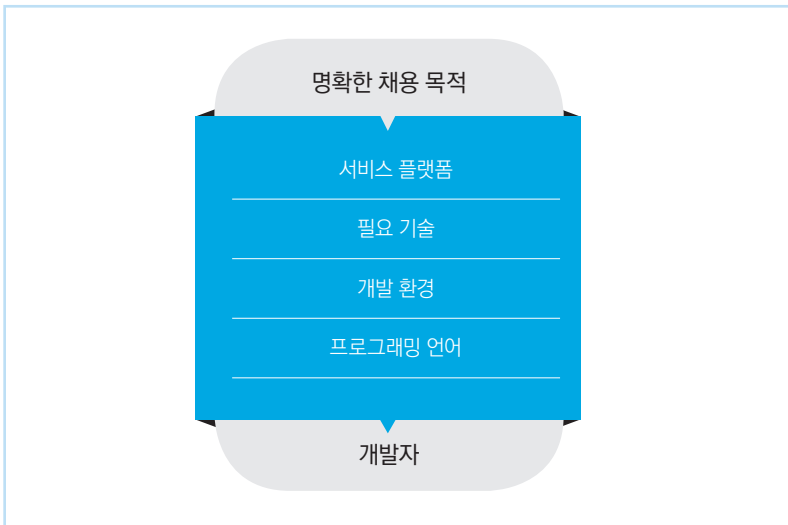
어느 기업이나 유능한 개발자를 뽑고 싶어 한다. 하지만 회사의 목적과 규모에 맞는 개발자 채용이 더욱 중요하다는 사실을 명심해야 한다. 현재 회사의 규모와 개발하려는 서비스 등을 명확히 파악한 후 채용을 시작해도 늦지 않다. 선불리 뽑은 후에 후회하는 것보다는 '천천히 가는 것이 더욱 빨리 가는 것'이라는 격언을 되새기며 채용에 임하면 좋을 것이다.

목적에 맞는 개발자 구분/채용 형태 정하기

그렇다면 위의 목적에 맞는 개발자를 채용할 때 무조건 높은 스펙과 경험을 가진 S급 혹은 A급의 경력자를 채용해야 할까? 아무리 경험이 많은 개발자를 채용한다고 하더라도 기업의 실제 업무에 적합하지 않다면 원하는 성과가 나오기 쉽지 않다. 그러므로 기업이 목적으로 하는 개발 범위를 살펴보고 무조건 고급(S급) 개발자가 아닌 현실적으로 해당 업무를 잘 수행할 수 있는 개발자를 뽑아야 한다. 회사 내 예산, 수행 업무, 업무 스타일 등 다양한 요소를 고려해서 채용해야 한다.

채용 목적을 명확하게 설정했다면 이제는 목적에 적합한 개발자를 구분하는 단계다. 구분하는 방법에는 두 가지가 있는데, 첫 번째는 분야별로 구분하는 것이고 두 번째는 등급별로 구분하는 것이다.

개발자를 구분하는 것은 매우 복잡하고 다양해서 명확히 정리하기 어렵지만, 이해를 돕기 위해 분야별로 간략히 구분하면 다음 그림과 같다.



분야별 구분 필터링

개발자 구분하기

- 개발자 채용을 위한 필터링
- 서비스 플랫폼 형태 정하기

우선 결과물에 해당하는 서비스 플랫폼 형태를 결정해야 한다.

크게 구분하자면 하드웨어 기반 서비스인지 소프트웨어 기반 서비스인지 구분해야 한다. '실시간 참고 재고 자동 관리 서비스'를 만들기 위해서는 공장 내 자동화 로봇 등 하드웨어와 이를 제어할 수 있는 소프트웨어 모두가 필요하다. 하지만 '공공 데이터 실시간 분석 및 시각화 서비스'를 만들기 위해서는 하드웨어보다는 소프트웨어 기반의 서비스가 필요하다. 기업에서 목적으로 하는 사업/프로젝트가 결정되었으면 이에 맞는 서비스 플랫폼(하드웨어/소프트웨어)을 우선적으로 결정해야 한다.

- **필요 기술 정하기**

다음으로 필요 기술을 정해야 한다. 해당 서비스의 구현을 위해서는 다양한 분야의 기술이 필요하다. 구체적으로 프론트 엔드, 백엔드/서버, 인공지능/빅데이터, 사물인터넷, 블록체인, 클라우드 등의 기술을 대표적으로 꼽을 수 있다. 최근에는 4차 산업 분야의 신기술을 학습한 인재들이 채용 시장에 많이 나오고 있다. 위 기술 중 회사의 목적에 맞는 서비스를 구현할 수 있는 기술을 결정해야 한다.

- **개발 환경 정하기**

다음으로 개발 환경을 결정해야 한다. 이클립스(Eclipse), 라자루스(Lazarus), 비주얼 스튜디오(Visual Studio), 파이참(PyCharm) 등 실제 업무에 활용하는 개발 환경의 종류는 매우

많다. 그중 회사 내의 개발자들이 주로 활용하는 개발 환경이 나 일반적으로 많이 사용하는 개발 환경을 우선으로 고려해 선택하는 것이 좋다.

- **프로그래밍 언어 정하기**

마지막으로 개발에 사용할 프로그래밍 언어를 정해야 한다. 개발 환경의 종류만큼이나 다양한 프로그래밍 언어가 존재한다. 서비스 플랫폼의 유형과 필요 기술, 그리고 개발 환경에 따라 프로그래밍 언어를 선정해야 한다. 대표적으로 C, C++, C#, JAVA, JavaScript, PHP, Python, Ruby, Perl 등이 있다.

위의 4단계를 거치면 목적에 적합한 개발자가 어떤 능력을 갖추어야 하는지 명확하게 알 수 있다. 위 내용이 명확하게 이해되지 않는 경우 회사 내 경력 개발자와 함께 해당 채용 목적을 명확하게 설정하는 편이 좋다.

- **개발자를 등급별로 구분하기**

소프트웨어 기술자 능력을 객관화하기 위해 2008년 소프트웨어 기술자 신고제(등급제)가 운용되었다. 그러나 등급 분류 기준 및 실무 경력 산정의 불합리성 등 개발자들의 수많은 원성을 사면서 2012년 11월에 결국 폐지되었다.

하지만 아직 현장에서는 이러한 등급을 활용하고 있는 곳이 많다.

실제 현장에서는 과거의 기준과 유사하게 초급/중급/고급/특급으로 구분하여 개발자를 채용한다. 또한, 소프트웨어산업협회에서는 매년 소프트웨어 기술자 평균 임금을 공표하는데, 2019년까지 공표된 내용은 기존의 소프트웨어 기술자 등급을 그대로 활용하였다. 물론 2020년부터는 분야별로 시간/일/월 평균 임금을 공표하는 방식으로 변경되었지만, 그전까지는 등급제가 여전히 활용되었다. 자세한 내용은 다음의 표를 참고하길 바란다.

표 4 | 2019년까지 활용된 소프트웨어 기술자 등급제

기술 등급	기술자격자	학력·경력자
기술사	·기술사	
특급 기술자	·고급기술자 자격 취득 후 3년 이상 소프트웨어 기술 분야에서 일정기간 경력을 갖추거나 근무한 사람	
고급 기술자	·중급기술자 자격 취득 후 3년 이상 소프트웨어 기술 분야에서 일정기간 경력을 갖추거나 근무한 사람 ·박사학위를 가진 자로서 기사자격을 취득한 자	
중급 기술자	·기사 자격을 취득한 자로서 3년 이상 소프트웨어 기술 분야에서 일정기간 경력을 갖추거나 근무한 사람 ·산업기사 자격을 취득한 자로서 7년 이상 소프트웨어 기술 분야에서 일정기간 경력을 갖추거나 근무한 사람 ·기사자격을 취득한 자로서 석사학위 취득 후 2년 이상 소프트웨어 기술 분야에서 일정기간 경력을 갖추거나 근무한 사람	
초급 기술자	·기사 자격을 취득한 자 ·산업기사 이상의 자격을 취득한 자	·전문학사 이상의 학위를 가진 자 ·고등학교를 졸업한 후 3년 이상 소프트웨어 기술 분야에서 일정기간 경력을 갖추거나 근무한 사람

기술 등급	기술자격자	학력·경력자
고급 기능사	·산업기사의 자격을 취득한 자로서 4년 이상 소프트웨어 기능 분야에서 일정기간 경력을 갖추거나 근무한 사람 ·기능사의 자격을 취득한 자로서 7년 이상 소프트웨어 기능 분야에서 일정기간 경력을 갖추거나 근무한 사람	
중급 기능사	·산업기사의 자격을 취득한 자 ·기능사의 자격을 취득한 자로서 3년 이상 소프트웨어 기능 분야에서 일정기간 경력을 갖추거나 근무한 사람	
초급 기능사	·기능사의 자격을 취득한 자	

표 5 | 2020년 신설된 소프트웨어 개발자 평균 임금

구분	일평균 임금	월평균 임금	시간평균 임금
① IT 기획자	403,081	8,424,393	50,385
② IT 컨설턴트	437,900	9,152,103	54,737
③ 정보보호컨설턴트	340,978	7,126,439	42,622
④ 업무분석가	501,090	10,472,778	62,636
⑤ 데이터분석가	335,799	7,018,209	41,975
⑥ IT PM	362,780	7,582,109	45,348
⑦ IT PMO	410,270	8,574,648	51,284
⑧ 소프트웨어 아키텍트	389,104	8,132,265	48,638
⑨ Infrastructure 아키텍트	461,684	9,649,203	57,711
⑩ 데이터 아키텍트	399,985	8,359,679	49,998
⑪ UI/UX 개발자	258,696	5,406,750	32,337
⑫ 응용 소프트웨어 개발자	305,985	6,395,094	38,248
⑬ 시스템 소프트웨어 개발자	247,970	5,182,563	30,996

구분	일평균 임금	월평균 임금	시간평균 임금
⑭ 임베디드 소프트웨어 개발자	271,214	5,668,383	33,902
⑮ 데이터베이스 운용자	274,324	5,733,364	34,290
⑯ NW 엔지니어	327,598	6,846,793	40,950
⑰ IT 시스템운용자	278,605	5,822,848	34,826
⑱ IT 지원 기술자	183,743	3,840,221	22,968
⑲ 소프트웨어 제품 기획자	426,419	8,912,158	53,302
⑳ IT 서비스 기획자	383,295	8,010,861	47,912
㉑ IT 기술영업	376,746	7,874,001	47,093
㉒ IT 품질관리자	402,554	8,413,382	50,319
㉓ IT 테스터	198,611	4,150,962	24,826
㉔ IT 감리	340,109	7,108,276	42,514
㉕ IT 감사	398,085	8,319,985	49,761
㉖ 정보보호관리자	376,529	7,869,458	47,066
㉗ 침해사고대응전문가	278,202	5,814,424	34,775
㉘ IT 교육강사	321,017	6,709,263	40,127

대기업/공공기관 등에서 사업을 발주할 때 어떤 역량을 가진 개발자가 몇 명 정도 필요한지 수치화를 통해 총사업비용 및 기간 등을 산정하곤 한다. 이때 별도의 분류 기준이 없으므로 과거에 사용하던 개발자 등급제를 기준으로 활용하는 곳이 많다.

회사에서 요구하는 개발 수준과 예산 등을 고려하여 필요한 개발자의 등급이 어느 정도인지 파악해야 한다. 해당 제도가 폐지되었다고 하지만, 아직도 공공기관 및 대기업에서 개발 용역 사업 발주시 예산

기준으로 사용되고 있다. 이런 등급제를 통해 우리 회사에 적합한 개발자의 유형 등을 파악할 수 있을 것이다.

채용 형태 정하기

채용을 위한 개발자를 구분하였다면 이제는 채용 형태를 정해야 한다. 우선 크게 구분하면 인소싱(Insourcing)과 아웃소싱(Outsourcing)으로 구분이 가능하다. 두 형태 모두 장단점이 분명하게 존재하므로 회사의 채용 목적에 맞는 채용 형태를 결정해야 한다.

● 인소싱(Insourcing) 활용하기

여러분이 고품질의 제품(서비스)을 만들어서 시장에 런칭하려고 한다고 가정해보자. 이를 개발할 수 있는 시간적인 여유와 비용도 있다고 하자. 이때 필요한 채용 형태는 무엇일까? 바로 인소싱을 이용하는 것이다. 인소싱은 해당 제품(서비스) 개발을 위해 전문인력을 구성하고, 이들을 활용하여 보다 높은 품질의 제품(서비스)을 만드는 것이다.

그렇다면 인소싱은 어떤 장점이 있을까? 인소싱은 아웃소싱에 비해 체계적인 품질 관리를 할 수 있다. 또한, 개발하는 직원들이 내부에 있으므로 회사에 대한 소속감이 높고 지속적인 의사소통이 가능하다. 개발팀을 이끄는 PM이 해당 분야에 대한 경험이 많다면 진행 속도 및 품질 관리는 더욱 잘 될 수 있다.

하지만 단점도 존재한다. 인소싱은 아웃소싱에 비해 더 많은 비용이 들어간다. 개발을 위한 여러 분야의 인력을 채용하고 관리해야 하기

때문이다. 한 사람의 직원(신규 졸업자)을 채용하여 정년까지 고용한다고 가정했을 때 발생하는 비용은 23억 4천만 원 정도라고 한다.

직원 채용은 보험, 복지 등 챙겨야 할 것들이 많으므로 적지 않은 비용이 든다. 또한, 내부 인력 간에 마찰이 생길 수 있으며 예상했던 것보다 일정이 지연될 수도 있다. 그러므로 이에 대한 지속적인 조율과 관리가 필요하다.

인소싱은 서비스 개발을 위한 시간과 비용에 여유가 있을 때 추천한다. 아웃소싱보다 훨씬 높은 품질의 결과물과 사후 서비스가 가능하다.

● 아웃소싱(Outsourcing) 활용하기

여러분이 스타트업을 창업하려고 한다. 창업 아이디어는 좋은데 개발할 수 있는 시간과 역량이 부족한 경우에는 어떤 선택을 해야 할까? 이때 필요한 채용 형태가 아웃소싱이다.

작은 규모의 스타트업이 전문 개발 인력을 보유하고, 서비스 전체를 개발하는 일은 쉽지 않다. 만약 ‘애완견 관리 애플리케이션’을 만든다고 해보자. 이 앱 하나를 제작하기 위해서는 서버 개발자, 앱 개발자, 디자이너 등 여러 명의 관련 인력이 필요하다. 하지만 신생 기업에서 이런 인력을 전부 뽑기는 쉽지 않다. 물론 이런 작업을 모두 담당할 수 있는 인력도 존재하나 이런 개발자를 찾기는 더 어렵다.

따라서 이런 유형의 스타트업에서는 아웃소싱을 선택하는 게 좋은 선택일 수 있다. 아웃소싱을 활용하면 어떤 장점이 있을까? 가장 큰

장점은 비용 절감이다. 또한, 완성에 대한 리스크와 시간도 절약할 수 있다. 인소싱의 경우, 우수한 개발자라 믿고 채용했는데 기대보다 역량이 부족할 수 있다. 또한, 협업이 쉽지 않은 성향일 수도 있다. 원하는 기간에 제품을 만들지 못하는 경우도 있다. 인소싱이 아닌 아웃소싱을 채택하면 이런 리스크를 현저히 낮출 수 있다. 아웃소싱 전문업체는 우리 회사보다 훨씬 많은 경험이 있으므로 원하는 서비스(제품)를 쉽고 빠르게 만들 수 있다.

그렇다고 좋은 점만 있는 것은 아니다. 우리가 기대했던 품질에 못 미치는 결과가 나올 수 있다. 또한, 아웃소싱 특성상 개발팀과의 소통이 어렵다. 해당 업무를 하는 팀이 외부에 있어 개선사항을 전달하기가 쉽지 않기 때문이다. 이런 문제가 발생하지 않도록 사전에 명확한 기준을 전달하고 이에 맞는 개발이 진행되는지를 지속해서 파악할 필요가 있다.

채용담당자 정하기

개발자 채용 공고를 내야 하는 단계에서 개발자와 채용담당자는 어떠한 역할을 하면 좋을까? 제일 좋은 방법은 채용담당자가 개발 관련 지식을 알고 채용을 진행하는 것이다.

그러나 일반적으로 개발을 잘 이해하고 있는 채용담당자는 흔하지 않다. 이로 인해 개발 경험이 전혀 없는 담당자가 개발자를 뽑는 아이

러니한 상황이 많은 기업에서 벌어지고 있다. 이러한 문제를 어떻게 해결하면 좋을까? 채용담당자에게 개발과 관련된 교육을 시키는 것이 제일 효율적이다. 개발자가 채용담당자의 업무를 할 수 있도록 직무를 바꿔주는 것도 좋은 방법이다. 하지만 어렵게 뽑은 개발자를 채용담당자로 바꾸는 일은 쉽지 않은 결정이다. 또한, 개발자는 생소한 인사 업무를 진행하기 위해 시간을 들여 공부해야 할 수도 있다.

개발을 잘 모르는 상태에서 채용을 진행하면 어떤 일이 벌어질까? 채용 공고를 내는 것조차 어려워질 수 있다. '프로그래밍 잘하는 직원을 뽑습니다'라고 공고를 낼 수는 없지 않은가.

구체적으로 공고를 내야만 좋은 개발자를 채용할 수 있는 확률이 높아진다. 다음 문제는 스크리닝(Screening) 단계에서 발생한다.

어떤 경력과 기술을 가진 개발자를 뽑아야 하는지, 이력서에 있는 내용은 무엇인지, 우리 회사에서 진행하는 업무와 유사한 프로젝트를 진행한 지원자는 누구인지 등을 기술적으로 알지 못하면 이력서에 쓴 내용을 이해하는 데 많은 어려움이 생긴다.

어떤 개발자가 이력서에 '나는 안드로이드 앱을 만들 수 있다', 'Java와 Kotlin은 다뤄본 적이 없다'고 썼다면 그의 실력을 의심할 수 있다. 안드로이드 앱은 Java와 Kotlin을 활용하여 개발하기 때문이다. 하지만 기술적인 내용을 모르면 해당 지원자를 무사통과시킬 수도 있다.

이렇게 채용이 진행되어 해당 지원자를 뽑았다고 생각해보자. 저 정도 역량을 가진 지원자가 개발 업무를 잘 수행할 수 있다고 생각하는가? 말기는 프로젝트마다 모두 엉망진창이 될 것이다. 결과적으로 다

른 개발자가 투입되어 마무리해야 하는 순간이 발생할 수 있다.

개발자 채용은 한두 명의 직원이 할 수 있는 업무가 아니라 협업을 통해 진행해야 하는 업무다. 여러 분야의 직원 간 협업을 통해야만 효율적인 채용 절차가 진행된다. 회사 내 직원들은 각자의 전문 분야를 활용해야 하고 정해진 역할도 다르다.

임원진(경영진)도 개발 및 IT 분야에 대해 알아야 한다

최근 매출 500억 정도 되는 쇼핑몰에서 개발자를 뽑지 못했다고 한다. 그 이유는 무엇일까? 쇼핑몰 대표가 IT 관련 지식이 없고 이로 인해 개발자가 회사에 크게 필요하지 않은 직군이라고 생각했기 때문이다. 또한, 이를 설득할 수 있는 IT 지식을 가진 경영진도 없었다. 그 결과 유능한 개발자 영입에 실패했고 좋은 서비스와 제품을 만들 기회를 놓쳐버렸다.

일반적인 IT 기업은 대표나 경영진이 관련 지식을 알고 있는 경우가 많으나, 비 IT 기업은 그렇지 못한 경우가 많다. 이러한 비 IT 기업에서 좋은 개발자를 뽑으려면 이들이 회사에서 다양한 의견을 제시할 때 들어줄 수 있는 환경이 되어야 한다. 개발자 입장에서는 개발팀도 꾸려지지 않은 비 IT 기업에 오는 것이 모험일 수 있다. 따라서 이들이 회사에 와서 구체적으로 어떤 성장을 할 수 있고 회사의 어떤 부분에 기여할 수 있는지를 설명해줄 수 있어야 한다. 즉, 비 IT 기업 내 주요 임원진 중 신뢰할 수 있는 사람이 개발 기술 혹은 최신 IT 기술을 배우는 것이 좋다.

회사 내 의사결정권을 가진 임원진이 IT 기술을 배우면 어떤 점이 좋을까? 우선 개발자를 뽑는 안목이 높아질 것이며, 이들을 적재적소에 배치하여 적극적으로 활용할 수 있을 것이다. 또한, 채용담당자나 개발자의 의견을 모두 청취한 후 회사의 전반적인 관점에서 판단과 결정을 내릴 수도 있을 것이다.

모든 임원진이 이러한 기술을 배우라는 것은 아니다. 비교적 IT 분야에 해박한 임원진 중 한 명이 학습하여 관련 지식과 경험을 축적하면 우리 회사에 좋은 개발자를 뽑을 기회가 늘어날 것이다.

채용담당자(인사팀)는 전체 전략 수립을 담당한다

채용담당자는 개발자 채용을 위해 다양한 전략을 수립해야 한다. 채용담당자는 외부의 다양한 요인을 파악하고 이에 따른 내부 자원 활용 방법 등에 대한 구체적 이해가 필요하다. 이를 위해 최신 기술 트렌드나 관련 내용을 어느 정도 숙지할 필요가 있다. 물론 채용담당자가 개발자만 채용하는 것은 아니므로 모든 내용에 대한 이해가 쉽지 않을 수 있다. 이럴 때 개발팀의 전문가와 협력하면 채용 전략 수립에 많은 도움을 받을 수 있다.

또한, 기업 전반의 경영 전략을 이해하고 이를 기반으로 한 인사 전략을 수립하는 노력도 필요하다. 우리 기업이 목표하는 서비스는 무엇인지? 이를 개발하기 위해 어떤 수준의 개발자가 필요한지? 등에 대해서 고민해야 한다. 이를 기반으로 인사 전략을 수립하면 채용에 많은 도움이 될 것이다.

채용담당자는 지원자에게 회사에 대한 긍정적인 이미지를 심어주는 역할도 해야 한다. 이들이 회사의 진심을 느낄 수 있도록 마음이 담긴 상호작용을 해야 한다. 채용 공고부터 채용 확정까지, 모든 과정을 신속하게 진행하고 지원자들에게 면접 결과 등에 대한 빠른 피드백을 제공해야 한다.

CTO/개발팀장/시니어 개발자는 실력자를 파악한다

이들은 채용에 결정적인 역할을 하는 사람들이다. 지원자들이 실제로 충분한 개발 실력과 경험을 보유하고 있는지를 판단해 줄 수 있다. 이력서를 검토해 자격과 코딩 스킬이 부족한 개발자들을 1차적으로 걸러주는 역할을 한다. 이들은 인터뷰에서도 큰 역할을 한다. 핵심적인 기술 분야 질문을 통해 지원자가 해당 역량을 실제로 가졌는지를 판단한다. 과거의 프로젝트 수행 시에 어떤 역할을 했는지도 파악할 수 있다.

우리 회사에 필요한 기술을 가장 정확하게 아는 사람은 시니어 개발자다. 이들을 통해 지원자의 개발 실력을 적극적으로 검증하는 편이 좋다. 또한, 이들은 수십 년간의 경력을 가지고 있으므로 관련 분야의 인맥이 풍부하다. 따라서 지원자를 최종 선발하기 전 레퍼런스 체크를 담당하여 진행할 수도 있다.

여기서 잊지 말아야 할 것이 있다. 채용 시기에는 해당 인력들의 다른 업무를 최대한 줄이고 채용에만 집중할 수 있는 환경을 만들어줘야 한다. 그래야 효율적이고 양질의 채용을 수행할 수 있을 것이다.

채용 채널의 다양화

요즘 기업들은 개발자 채용이 정말 어렵다고 말한다. 주변을 보면 유능한 개발자들이 많이 있는데 왜 개발자를 채용하는 일이 이렇게 힘들까? 우리가 뽑고자 하는 개발자들은 어디에 있는 것일까? 그 이유를 알아보자.

개발자는 적극적으로 구직활동을 하지 않는 경우가 많다

스택 오버플로(Stack Overflow)는 컴퓨터 프로그래밍과 관련된 여러 주제에 대한 질문과 답변을 할 수 있도록 제작된 웹사이트이다. 이 사이트에서 전 세계 개발자 10만 명을 대상으로 구직활동에 대한 설문 조사를 진행했다. 설문의 응답자 중 16%만이 구직활동을 적극적으로 하고 있다고 답했다. 반면 응답자의 23%는 새로운 구직활동에 큰 관심을 보이지 않는 것으로 나타났다. 또한, 응답자의 62%는 구직활동을 하지는 않지만 지금의 직장보다 좋은 조건이 보장되는 제안이 온다면 이직할 의사가 있다고 답변하였다. 해당 설문 결과만 살펴봐도 우리가 원하는 개발자들을 찾기 힘든 이유를 알 수 있다.

그렇다면 우리는 위에서 언급한 '적극적 구직활동을 하는 개발자(16%)'와 '좋은 조건이 있다면 이직을 할 의사가 있는 개발자(62%)'에게 좋은 개발자를 찾고 있다는 것을 알려야 하지 않을까? 이를 효과적으로 알리는 방법에는 어떠한 것들이 있을까?

채용 채널은 최대한 다양한 방법을 활용한다

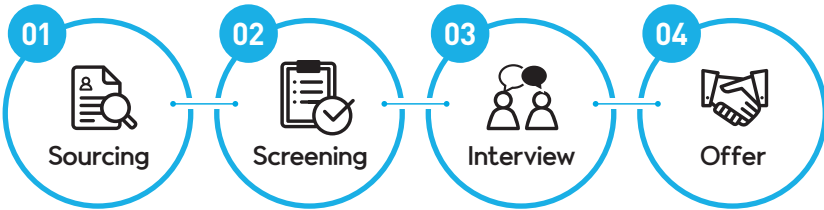
앞 장에서 언급한 것처럼 개발자들에게 우리 회사의 채용을 알리기 위해 다양한 채널을 활용해야 한다. 앞서 다룬 내용이므로 간략하게만 소개한다.

회사 내부에서 직접 채용하는 경우 소셜 미디어, 구인/구직 포털, 세미나/콘퍼런스/박람회, 개발자 커뮤니티 등을 통한 홍보를 진행할 수 있으며, 사내 직원 추천제도를 활용하기도 한다. 외부 업체에 채용을 의뢰하는 경우에는 헤드헌팅 업체에 채용을 의뢰하는 방법, 소프트웨어 분야 HR 기업에 채용을 의뢰하는 방법 등이 있다.

여기에서 요점은 한두 가지 채널에만 한정하면 안 된다는 점이다. 최대한 많은 채널을 통해 유능한 개발자가 우리 회사의 채용에 관심을 가지고 지원할 수 있는 기반을 마련해야 한다. 번거롭다는 이유로 이를 게을리한다면 좋은 개발자는 다른 회사로 가버릴 수 있다.

02

채용 실행하기



Sourcing



채용 공고는 명확히 하라

미혼 남녀가 결혼 정보 회사를 이용할 때 어떤 방법을 통해야 좋은 상대를 만날 수 있을까? A양은 “저는 젊고 연봉이 높으며 사회성이 좋은 분을 만나고 싶어요”라고 의뢰했다. B양은 “저는 대기업에 근무하는 00세 정도의 나이에 연봉 수준은 0천만 원 이상이고, 동아리 활동을 1개 이상 하는 남성분을 만나고 싶어요.”라고 의뢰했다.

여러분이 결혼 정보 회사 직원이라면 누구의 요청을 더 잘 들어줄 수 있을까? 당연히 후자가 명확한 조건을 제시했기 때문에 원하는 상대를 만날 확률이 높다.

채용도 이와 마찬가지로이다. 미래를 함께할 배우자를 선택하는 게 중요한 것처럼 개발자를 채용하는 일도 배우자를 고르는 것 이상으로 중요하다. 기업에게 있어 채용은 결혼과도 같다. 배우자를 잘못 만나면 평생을 고생하듯이 개발자를 잘못 뽑으면 골머리를 썩게 된다.

그렇다면 개발자 채용의 시작은 어떻게 해야 할까? 우선은 우리 회사에서 개발자를 채용하는 것을 알려야 하므로 명확한 채용 공고부터 만들어야 한다. 여기서 강조할 것은 ‘명확한’ 채용 공고라는 점이다. 막연하게 올린 채용 공고는 그만큼 좋은 개발자를 채용할 확률을 떨어뜨린다. 대부분의 채용 공고에 해당하는 내용이겠지만 특히 개발자 채용에서는 명확한 내용의 직무 기술서(Job Description, 이하 JD)를 올리는 것이 좋다. 명확한 JD는 다수의 지원자 및 높은 스펙의 지원자를 확보할 수 있는 가장 큰 무기이고, 직무 연관도가 낮은 지원자들이 불필요하게 지원하는 것도 방지할 수 있는 최소한의 안전장치다. 명확한

JD를 작성하는 요령을 자세히 소개한다.

- 누가 쓰면 좋은가?

JD 작성을 할 때 어디부터 시작해야 할지 막막한 경우가 많다. JD 작성은 반드시 해당 업무를 가장 잘 알고 있는 사람(개발자)에게 도움을 받아야 한다. 회사에 HR팀이 있다면 해당 팀과 개발자들이 협업하여 작성하게 하면 금상첨화다. 만약에 HR팀에게 JD 작성을 모두 맡긴다면 해당 직원들은 개발 관련 지식이 없는 경우가 많으므로 좋은 JD가 나올 수 없다.

- 어떤 내용을 써야 하나?

추상적인 업무를 기술하는 것이 아니라 실제 해야 할 업무를 구체적으로 작성하는 것이 좋다. JD에 반드시 들어가야 하는 항목들은 아래와 같다.

- **제목:** 회사에서 요구하는 기술과 경험을 한 문장으로 설명할 수 있으면 좋다.
- **회사 소개:** 우리 회사가 추구하는 가치와 기업 문화를 상세히 소개한다. 회사의 성장과 개인의 발전이 연계되는 회사, 협업과 소통이 활발한 회사, 젊은 에너지가 넘치는 회사, 다양한 의견을 포용할 수 있는 회사 등과 같은 여러 장점을 강조해야 한다. 또한, 해당 직무를 통해 기대할 수 있는 개인적인 발전 요소, 전망, 진로

등을 명시한다.

- **고용 기간 및 고용 형태:** 정규직, 계약직과 같이 명확한 고용 형태를 명시할 필요가 있다. 계약직 모집 시에는 고용 기간을 함께 기재한다.
- **담당 업무:** 업무의 배경은 상세하게, 담당 업무는 명확하게 기술해야 한다. 어느 부서에서 어떤 개발 업무를 담당할지 최대한 구체적으로 작성하는 것이 좋다. 해당 업무가 기존 사업을 개선하는 작업인지, 신규 사업을 위한 것인지, 특정 플랫폼 혹은 모듈 제작인지 등을 구체적으로 명시하는 것이 좋다. '신규 고객 서비스 개발 및 대응 업무'라는 추상적 표현보다는 '머신러닝 알고리즘을 활용한 자사 콘텐츠 추천 서비스 개발 및 알고리즘 개선 업무'라는 구체적 표현을 활용할 필요가 있다.
- **필요 기술(자격 요건):** 서비스 플랫폼(하드웨어/소프트웨어), 필요 기술(인공지능, 빅데이터, 사물인터넷), 개발 환경(이클립스, 파이썬), 활용 가능한 언어(Java, JavaScript, HTML, CSS) 등 최대한 구체적으로 기술해주는 것이 필요하다. '웹서비스 개발을 잘하는 사람'이라는 추상적 표현보다는 'HTML/PHP/JavaScript를 활용한 웹 개발 업무 경험을 3년 이상 보유한 사람'이라는 구체적 표현을 활용하는 것이 좋다. 해당 부분을 작성할 때는 최소한 현재 해당 업무를 담당하고 있는 개발자의 리뷰를 받을 필요가 있으며, 가급적 이들과 협업하여 작성하는 것이 좋다. 기술의 빠른 변화 탓에 채용담당자가 해당 내용을 정확하게 인지하지 못하는 경우도 많기 때문이다.

- **근무지:** 근무지의 위치는 이직을 결정하는 가장 중요한 요소다. 따라서 근무지의 정확한 위치와 근무 형태(예: 주 1회 재택 근무, 주 4회 근무)를 소개한다.
- **연봉 및 보상 체계:** 지원자들이 가장 중요하게 생각하는 요소인 연봉 수준을 대략적으로 명시하는 것이 좋다(예: 개발자 최저 연봉 6천 만 원). 또한, 다양한 보상 체계가 있음을 명시하라(예: 성과급, 명절 상여금).
- **복지:** 업무에 필요한 도서 구매 지원, 세미나 및 교육 참가비 지원, 개인 사무실 제공, 식사 및 간식 제공, 통근 버스 제공, 건강 검진 지원, 학비 지원, 자율 출퇴근 등 회사에서 제공하는 복지 내용은 상세하게 기술할 필요가 있다. 생각보다 많은 개발자들이 이러한 환경을 중시한다.
- **우대 사항:** 다양한 프로젝트 경험, 특정 개발 언어 사용 경험, 오픈 소스 contribution 경험, 특징 직무 관련 경험, 수상 경험 등의 다양한 우대 사항을 명시한다.

● JD 작성 팁

JD만 매력적으로 기술해도 회사에 적합한 개발자를 충분히 뽑을 수 있다. 이 과정이 너무 번거롭고 힘든 과정인 것을 알고 있다. 하지만 JD를 잘못 작성하면 우수한 개발자를 모시는 일은 힘들다고 보면 된다. 좋은 JD 작성을 위한 팁 몇 가지를 소개한다.

첫째, 우리 회사 개발 조직의 모든 정보를 투명하게 공개해야 한

다. 앞서 기술한 모든 요소가 다 들어가야 하며 내용은 상세하게 기술할수록 좋다. 요소별로 제목 정도만 쓰는 것이 아닌 부연 설명이 충분히 담겨야 한다. 'Frontend와 Backend 개발자 각 1명을 채용하며 이들 간의 협업 능력 중요'라고 기술하는 것보다는 'Frontend와 Backend 개발자 각 1명을 채용합니다. 개발팀 내에서는 Frontend와 Backend 개발자를 크게 구분하지 않습니다. 프로젝트가 발생하면 세부 주제를 나누어 선호하는 부분을 각자 맡아서 개발을 진행합니다. 개발이 완료되면 코드 리뷰 시간을 가지므로 서로 성장할 수 있는 분위기가 형성됩니다'와 같이 아주 구체적으로 적는 것이 우리 회사에 맞는 개발자를 뽑는 지름길이다.

둘째, JD만으로 우리 회사의 색깔이 묻어날 수 있도록 작성하라. 회사 이름이 비어있는 상태에서 JD를 읽어도 '이 회사는 OO이구나'라고 알 수 있도록 작성해야 한다. 다른 회사의 JD를 참고하는 것은 좋지만 그대로 갖다 쓰는 것은 바람직하지 못하다. 우리 회사만의 문화, 분위기, 개발 환경 등을 구체적으로 반영하여 JD 하나만으로 회사를 느낄 수 있도록 만들자. OO은행의 '빅데이터 처리' 업무 JD를 쓴다고 가정해보자. 단순히 '빅데이터 처리'라고 하면 어떤 업무인지 전혀 감이 오지 않는다. 하지만 여기에 'OO은행에서는 고객의 송금 및 입출금 데이터를 기반으로 은행 서비스의 위험도를 감지하여 금융 사고를 방지하고 있습니다. 따라서 실시간 고객 금융 빅데이터를 분산 방식을 통해 처리하여 금융 서비스의 품질 향상에 도움을 주실 개발자를 찾고 있습니다'라고 표현하면 은행 업무 및 서비스에 대한 색깔이 묻어

난다. 이렇게 만들면 회사에 적합한 개발자가 지원할 수 있도록 도움을 줄 수 있다. 또한, 우리 회사에 대해 구체적으로 알리는 방법이기도 하다.

셋째, 개발자 집단에 대해 이해하고 JD를 작성하라. 개발자 집단은 일반 직장인과는 다른 그룹이다. 앞서서도 소개했지만 개발자의 문화 및 최근 트렌드 등을 명확하게 읽고 JD를 작성하는 것이 좋다. 수년 전에 활용했던 JD는 이미 시대에 뒤쳐졌다. 따라서 최근 개발자 채용 트렌드를 읽고 이에 맞춘 JD가 필요하다. 자세한 내용은 ‘개발자, 당신은 누구인가?’ 부분을 참고하면 이해가 쉬울 것이다.

좋은 개발자는 내부 직원이 더 잘 알고 있다

채용 공고를 통해 우수한 개발자를 바로 채용하면 좋겠지만 현실적으로 쉽지 않은 경우가 많다. 이때 활용할 수 있는 좋은 방법은 내부 직원 추천제이다. 사내 직원의 주변에서 우수한 개발 인력을 찾아 추천받는 채용 방법이다. 손꼽히는 해외 IT 기업들은 직원 추천제를 적극적으로 활용한다. 심지어는 직원 추천을 통해야만 제대로 된 면접을 볼 수 있다고 한다. 그렇다면 직원 추천제를 활용하면 어떠한 장점이 있을까?

가장 큰 장점은 채용을 위한 소요 기간이 획기적으로 단축된다는 점이다. 외부 연구 결과에 따르면 직원 추천제를 통해 입사하는 직원들의 평균 채용 기간은 29일 정도지만, 자사의 채용 사이트를 통해 입사하는 직원들의 평균 채용 기간은 39일이 걸리는 것으로 알려졌다.

둘째, 비용 효율성 측면에서 우수하다. 외부 HR 회사를 활용하게 되면 적지 않은 수수료를 지불해야 한다. 반면에 채용 성과로 임직원에게 지급하는 보너스는 수수료에 비하면 낮은 편이다.

셋째, 회사에 대한 정보를 충분히 알고 입사할 수 있다. 최근 자료에 따르면 신입사원 3명 중 1명이 입사 1년 이내에 조기 퇴사를 하는 것으로 조사되었다. 퇴사 사유를 보면 '직무가 적성에 맞지 않아서'라고 답한 사람이 절반 이상(51.4%)을 차지한다.

사전에 입사할 기업에 대한 정보를 충분히 얻는다면 퇴사 확률은 현저하게 낮아진다. 개발자에 대한 처우를 어떻게 하는지, 회사 분위기는 어떤지, 복지는 괜찮은지 등의 정보를 직원을 통해 미리 알고 들어오기 때문에 공채로 뽑은 개발자보다 이직이나 퇴사 확률이 낮아진다.

넷째, 회사 적응을 빠르게 할 수 있다. 신규 혹은 경력 개발자로 입사할 경우 회사 분위기나 문화에 적응하는데 생각보다 많은 시간이 걸릴 수 있다. 개발자들의 성향에 따라 이 기간은 더 길어질 수도 있다. 이런 문제로 회사에서는 입사자가 수월하게 적응할 수 있도록 멘토링 등의 제도를 운용하기도 한다. 직원 추천제로 입사할 경우 추천한 직원이 입사자가 잘 적응할 수 있도록 도움을 주는 멘토 역할을 수행할 수 있다는 장점이 있다.

직원 추천제가 빛을 발하려면 어떻게 해야 할까? 정답은 회사 내부에 있다. 즉, 내부 직원을 잘 관리하고 아껴야 한다. 직원 본인이 회사에 만족해야 적극적으로 추천할 수 있기 때문이다. 대우가 나쁘고 분

위기가 영망인 회사라고 느낀다면 지인에게 입사를 추천할 수 있을까? 그럴 확률은 거의 없다고 보면 된다. 그러므로 회사의 소중한 자원인 개발자들이 만족할만한 개발 환경, 복지제도, 사내 분위기 등을 만들 수 있도록 노력해야 한다.

헤드헌팅(서치펌)을 적극적으로 활용하라

기업 대표가 좋은 개발자를 Sourcing하기 위해 발로 뛰면 좋겠지만 쉽지 않은 것이 현실이다. 이러한 어려움을 해결할 수 있는 곳이 헤드헌팅 업체와 서치펌이다. 헤드헌팅 업체에 원하는 스펙의 개발자 정보를 전달하고 이에 맞는 채용을 의뢰할 수 있다. 다만 헤드헌팅 업체는 이 분야에 대한 전문성이 비교적 낮아 유능한 인재를 걸러내기가 쉽지 않을 수 있다. 다음으로 서치펌에 채용을 의뢰할 수 있다. 이 기업들은 소프트웨어 개발 분야에 대한 전문성을 가지고 있으므로 보다 전문적인 채용 업무를 수행할 수 있다. 관련 인맥 등 다양한 요소에서 우위를 선점하고 있어 개발자 채용 시에 유용하게 활용할 수 있다. 또한, 서치펌을 통해 채용 인력에 대한 전문가들의 검증 절차를 거칠 수 있다는 장점도 있다.

서치펌을 이용하면 좋은 개발자를 채용하기 위한 기회비용도 절약할 수 있다. 물론 서치펌에 지불하는 수수료가 있겠지만 우수한 인재의 영입으로 회사에 발생할 이익을 생각하면 전혀 큰 비용은 아니라고 생각한다.

물론 모든 서치펌이 이런 업무를 잘할 수 있는 것은 아니다. 개발

자 채용에 전문성과 충분한 이력을 가진 업체를 선정해야 한다. Tech HR을 전문적으로 하는 서치펌을 잘 선정해야 하며 전문적인 검증 없이 이력서만 전달하는 업체는 피해야 한다. 서치펌을 활용하기로 했으면 적극적으로 컨설팅을 받을 필요가 있다. 즉, 우리 회사가 어떤 인재를 뽑으려고 하는지 최대한 자세하게 설명하는 것이 좋다. 서치펌은 구직자와 구인회사의 모든 접점을 가지고 있기 때문에 상세한 설명을 기반으로 최대한 적합한 인재를 선별해 줄 것이다.

Screening

명확한 목적을 가진 채용 공고 또는 직원 추천 등의 방법을 통해 지원자들을 모았다면 더욱 중요한 단계가 기다리고 있다. 바로 뛰어난 개발자들을 고르는(Screening) 단계이다. 우리는 대면 면접으로 지원자를 심사숙고하여 골라내야 한다.

이 단계가 빠르게 종료될수록 채용에 소모되는 시간을 단축할 수 있다. 우수한 지원자만을 제대로 골라 면접 단계로 간다면 자격이 부족한 개발자를 대상으로 인터뷰하는데 쏟는 시간을 줄일 수 있다.

이력서를 통해 기초적인 정보를 파악하자

많은 이력서를 검토하다 보면 이 지원자가 우리 회사에 대해 얼마나 잘 이해하고 지원서를 작성했는지 빠른 파악이 가능하다. 훌륭한

지원자들은 기업에 대해 다양한 조사 및 철저한 분석을 통해 이력서를 쓴다. 경험이 풍부한 채용담당자들은 이력서 몇 줄만으로도 이를 파악할 수 있다. 때문에 이력서 검토에 많은 시간과 노력을 쏟아야 하며 이를 효율적으로 수행할 수 있는 인력이 필요하다. 사내에 많은 경험을 가진 인사담당자와 개발자가 없다면 차라리 외부의 전문가 혹은 HR 전문업체에 의뢰하는 편이 낫다. 비용을 이유로 전문성과 경험이 없는 내부 인력만을 고집한다면 오히려 잃는 것이 더 많다.

그럼 이력서는 어떻게 검토하면 좋을까? 먼저 채용 직무 기술서를 통해 공고했던 자격과 기술을 꼼꼼하게 찾아내야 한다. 지원자 중 우리 회사에서 필요로 하는 기술을 보유한 개발자를 잘 골라내야 한다. 이 과정에서 꼼꼼하게 확인하지 않아 인재를 놓친다면 이보다 더 안타까운 일은 없다.

정규직 채용이라면 이직률도 고려해야 한다. 지원자가 현 직장(혹은 이전 직장)에서 어떤 업무를 수행했으며 이직을 원하는 구체적인 사유도 파악해야 한다. 우리 회사로 이직한다는 것은 반대로 말하면 얼마 지나지 않아 다른 회사로 이직할 수 있는 확률도 있다는 얘기다.

현 직장(혹은 이전 직장)의 근무 경력이 짧다면 이직의 구체적인 이유를 알아내야 한다. 현 직장에서 습득한 기술이 무엇인지, 어떤 프로젝트에 참여했고 맡은 임무는 무엇이었는지 등의 정보를 구체적으로 파악하는 편이 좋다.

다양한 직무 관련 경험을 파악하는 일도 중요하다. 만약에 우리 회사에서 진행하려는 프로젝트와 유사한 프로젝트를 진행한 경험이 있

다면 금상첨화이다. 하지만 비슷한 프로젝트라도 세부적인 내용은 다를 수 있다. 그러므로 해당 프로젝트에서 어떤 기술을 활용하여 어떤 업무를 진행했는지, 그리고 어떤 임무를 수행했는지를 구체적으로 파악하는 것이 중요하다.

꼭 필요한 일만 했는지, 아니면 다양한 분야의 경험을 쌓기 위해서 노력했는지를 파악하는 것도 중요하다. 학교나 회사에서 반드시 해야 하는 일 위주로 프로젝트 경험을 쌓은 지원자들이 있는가 하면 스스로 재미를 느끼거나 취미로 프로젝트를 수행한 지원자들도 있다. 정말 열정적인 지원자들은 이러한 경험을 다수 보유하고 있는 경우가 많다. 단, 이런 지원자라도 해당 경험의 진실성에 대해서 깊이 있게 검토할 필요가 있다. 이력서를 화려하게 꾸미기 위해 다수의 프로젝트를 문어 발식으로 만들 수도 있기 때문이다.

개발자의 기술 검증하기(코드 리뷰/코딩 테스트)

최근 개발자 채용에서는 면접 전 코딩 테스트가 필수 요소로 자리 잡는 추세이다. 국내외 IT 기업인 구글, 네이버, 카카오 등은 개발자를 채용할 때 필수적으로 코딩 테스트를 실시한다. 개발자들은 테스트에 응시하여 일정 점수 이상을 획득해야만 취업할 기회가 주어진다. 심지어는 스펙을 보지 않고 코딩 테스트와 면접만으로 채용하기도 한다. 그만큼 실력을 객관적으로 검증할 수 있는 코딩 테스트가 중요하다는 의미로 해석된다. 개발자 채용 시에 코딩 테스트를 하지 않는 것은 정말 위험한 일이다. 실력이 부족한 개발자가 입사해 바로 성과가 나와

야 하는 프로젝트를 제시간에 마치지 못하거나 완성도가 낮은 결과물을 냈을 때의 피해는 상상할 수 없을 정도로 크다.

● 코딩 테스트, 어떻게 할 것인가?

코딩 테스트를 할 때 명심할 점이 있다. 만약 A라는 기술이 필요하다면 해당 기술에 대한 검증만 진행하는 것이 좋다. 즉, 검증이 필요한 영역만을 한정하여 테스트하고 불필요한 것을 테스트하는 데 시간을 낭비하지 말아라.

코딩 테스트는 어떻게 준비해야 할까? 회사에 따라서 조금씩 다르겠지만 공통으로 뽑아낼 수 있는 3가지 단계가 있다. 첫째, 우리 회사에서 하는 일 또는 채용의 목적을 명확히 분석하라. 코딩을 통해 어떤 문제 혹은 프로젝트를 해결할 것인지, 버그 픽스 혹은 유지보수를 할 것인지 등의 목적을 찾아내야 한다. 둘째, 이것을 기반으로 대표 케이스를 뽑아낸다. 실제 업무에서는 개발이 아니라 개발된 내용을 수정하는 일이 더 많을 수 있다. 이를 위한 방법의 하나로 회사 내 버그 중 하나를 샘플링해서 풀게 하는 것도 좋은 시험이다. 셋째, 이러한 대표 케이스를 활용하여 사내에서 테스트를 진행한다. 회사 내 뛰어난 개발자와 평범한 개발자를 몇 명 뽑아서 대표 케이스 문제를 풀도록 하고 결과를 확인한다. 해당 문제를 푸는 과정에서 질문이 많이 발생하는 부분은 실제 채용 시 수정하여 진행한다. 수정하지 않고 이 부분을 면접에서 활용하는 것도 좋은 방법이다.

그렇다면 우리 회사가 뽑는 목적에 맞는 테스트는 어떻게 진행하는

것이 좋을까? 코딩 테스트는 크게 나누어 초급, 중급, 고급 역량을 기준으로 테스트를 할 수 있다. 채용을 원하는 목적에 따라 이에 맞는 테스트 방법을 정하면 된다. 초급은 주어진 문제를 잘 해결할 수 있는가에 초점을 맞추면 된다. 즉, 일반적인 수준의 문제를 주고 주어진 시간 내에 얼마나 빠르게 잘 해결하는가를 평가하면 된다. 중급은 얼마나 효율적으로 문제를 해결하는가 또는, 최적화를 얼마나 잘 수행하는가를 평가하면 된다. 초급 개발자를 평가할 때보다 훨씬 세세한 부분을 평가해야 하므로 경력이 풍부한 개발자들을 위주로 평가 위원을 구성하는 것이 좋다. 고급 개발자 채용은 설계 능력을 중점적으로 평가해야 한다. 단순한 코딩 테스트로 이것을 평가하기는 쉽지 않다. 고급 역량을 가진 개발자를 보면 초, 중급 테스트에서는 월등한 산출물을 보여주지 못했지만 설계 능력은 이를 초월한 역량을 보유하고 있는 경우도 있다. 고급 개발자 채용 시에는 코딩 테스트에만 의존할 것이 아니라 해당 개발자가 작성한 깃(Git) 저장소 등을 적극 활용하여 심층적으로 역량을 평가하는 것이 중요하다.

● 코딩 테스트만이 최고의 검증 방법일까?

코딩 테스트 결과 정답을 잘 맞히더라도 코드의 퀄리티 자체가 좋지 않은 지원자가 있을 수 있다. 정답은 틀렸지만 좋은 퀄리티의 코드를 작성한 지원자가 있을 수도 있다. 해당 지원자가 어떤 방식으로 문제에 접근하는지, 그리고 어떻게 소통하고 풀어나가는지를 파악하는 것이 중요하다. 무조건 코딩 테스트를 못 봤다고 해서 인터뷰 대상으로 올리

지 않는 것은 좋지 않다. 코딩 혹은 문제를 해결하는 과정을 잘 살펴보고 우리 회사의 목적에 적합한지를 파악하는 것이 중요하다. 이를 위해 해당 기술을 잘 알고 있는 전문 개발자에게 도움을 요청하는 것이 좋다. 좋은 코드는 해당 분야의 전문가만이 파악할 수 있으며 일반 개발자들이 쉽게 분석할 수 있는 수준은 아니기 때문이다.

한 가지 더 추천하고 싶은 방법은 WebEx, IM, typewith.me와 같은 사이트를 활용해 지원자가 코드를 작성할 때 풀어나가는 과정을 실시간으로 보는 것이다.

코딩 문제를 풀었다고 해서 우리 회사에서 주어지는 다양한 문제 상황을 해결할 수 있다는 보장은 없다. 따라서 다음 단계에서 이어지는 인터뷰를 통해 지원자가 실제로 코딩을 잘 수행할 수 있는 역량을 보유하고 있는지를 파악해야 할 것이다.

- 코딩 테스트에도 한계점이 있다

개발자의 역량을 객관적으로 평가할 수 있는 코딩 테스트이지만 한계점도 분명 존재한다. 첫째, 테스트를 접하게 되는 개발자들이 이를 선호하지 않으며 이에 부담을 느껴 지원 자체를 꺼리는 일도 충분히 발생할 수 있다. 둘째, 작은 규모의 기업이나 비 ICT 기업의 경우 해당 테스트를 자체적으로 진행하는 게 쉽지가 않다. 결국 전문 회사에 외주 형태로 진행할 수밖에 없는데 이를 위해 적지 않은 비용을 지불해야 한다. 회사에서 반드시 필요로 하는 분야/기술에 대한 심층적인 테스트 또한 만만치 않다. 만약에 가능하더라도 커스터마이징을 위해

많은 비용을 들여야 하므로 현실적으로 어려운 문제가 발생한다. 또한, 외주 업체에 해당 업무를 위탁했을 때 부정행위나 기출문제가 유출되는 등의 공정하지 못한 테스트가 될 가능성도 있다. 셋째, 테스트 대부분이 교과서적인 내용이 많아 실제 현장에서 코딩을 수십 년 한 유능한 개발자라도 시험 응시를 위해서는 다시 공부가 필요하다. 우리가 아무리 한국말을 유창하게 한다고 해도 한국어 능력 시험을 위해서는 별도의 공부가 필요한 것과 같은 원리이다.

Interview

면접 준비하기

이제 우리 회사에 입사할 개발자를 만나는 시간이 다가왔다. 우리는 내실 있는 인터뷰를 위해 많은 것을 준비할 필요가 있다. 지원자가 우리 회사를 알기 위해 노력을 한 만큼 우리도 지원자를 맞을 준비를 정성껏 해야 한다. 그래야 우수한 개발자를 채용할 수 있다.

- **면접 전 지원자를 꼼꼼하게 검토하라**

면접 전에 지원자를 대상으로 검토해야 할 사항들을 빠짐없이 숙지하여 불필요한 시간 낭비를 막는다. 채용 접수 후 인터뷰까지, 회사 내 부에는 여러 단계가 있으므로 생각보다 시간이 오래 걸린다. 지원자들은 이 사이에 우리 회사뿐 아니라 다른 회사에도 지원했을 확률이 높

다. 따라서 다른 회사로 이직을 결정한 것은 아닌지, 아직도 우리 회사에 입사할 의향이 있는지 확인할 필요가 있다. 어떤 지원자는 다른 회사로부터 합격 통보를 받았거나 채용이 진행 중일 수도 있다. 해당 정보를 빠르게 파악할 수 있으면 좋겠지만 지원자는 타사로의 이직이 확정되기 전까지는 절대 공개하지 않는다. 따라서 면접 전 우선으로 해당 내용을 어느 정도는 파악해 면접 의사가 없는 지원자는 참석을 강요하지 않는 것이 좋다.

다음으로 지원자가 원하는 복지와 보상은 무엇인지 구체적으로 파악하는 것이 필요하다. 금전적인 내용을 중시하는지, 근무환경을 중시하는지 등 개발자마다 추구하는 가치는 다르다. 지원자가 중시하는 가치가 무엇인지 사전에 파악한다면 많은 도움이 될 것이다. 본인의 가치와 전문성을 키울 수 있는 회사를 원하는 지원자라면, 이에 맞는 가치와 인터뷰 질문을 준비하는 것이 바람직하다.

지원자 선발 시 검토해야 할 것 중 하나는 출퇴근 거리다. 원거리에 사는 지원자의 경우 입사하면 회사 근처로 이사를 오겠다고 말하곤 한다. 하지만 정작 입사 후에는 그렇지 못할 가능성이 크다. 그러면 결과적으로 근무 만족도가 저하되어 다시 다른 회사로의 이직을 고민하는 상황이 발생한다. 따라서 지원자의 거주지가 통근이 가능한 수준인지 확인해야 한다. 최근에는 많은 기업들이 재택 근무를 도입해 활용하고 있다. 각자의 기업 문화에 따라 해당 부분에 대한 가중치를 고려하면 된다.

- **전화(화상) 인터뷰를 할 경우 명확한 기준을 설정하라**

지원자의 상황이 여의치 않거나 대면 인터뷰 전에 사전 검증을 하고 싶다면 전화(화상) 인터뷰도 좋은 방법이다. 화상 인터뷰보다는 전화 인터뷰가 서로 부담이 덜하면서 자연스러운 대화를 나누는 방법이라고 생각한다. 전화로 인터뷰를 진행하면 효과가 다소 떨어질 것으로 생각할 수 있다. 하지만 실제로 인터뷰를 진행해보면 짧은 시간 내에 효율적으로 지원자를 파악할 수 있다. 지원자는 면접 전 회사에 대해 한층 더 친근하게 느낄 수 있는 계기가 되므로 회사와 지원자 모두에게 좋은 방법이다.

전화 인터뷰는 너무 길게 하는 것보다는 30분 이내의 시간을 갖고 핵심적인 내용 위주로 진행하는 게 좋다. 전화 인터뷰 후에 지원자에 대한 확신이 들면 대면 인터뷰 일정을 잡는다. 즉, 서류 전형 후 전화 인터뷰를 진행하고 적합한 지원자로 판단되면 실제 면접으로 이어지는 패턴이다.

여기서 주의할 점이 있다. 단순하게 지원자의 느낌을 파악하는 정도에서 그치면 안 된다. 전화 인터뷰를 진행할 때에도 구체적인 기준을 잡고 객관적인 인터뷰를 해야 한다. 또한, 지원자가 최대한 편하게 참여할 수 있도록 소요 시간, 주제 등에 대해서 미리 알려주는 배려도 필요하다.

면접 진행하기

이제 면접 대상자를 만날 차례이다. 면접은 지원자와 회사, 모두가

긴장하는 순간이다. 그만큼 중요한 과정이니 꼼꼼하게 준비할 필요가 있다.

● 지원자가 우리 회사와 적합한지 파악하자

지원자는 본인이 쓴 이력서를 잘 기억하고 설명할 수 있어야 한다. 이력서 작성에 많은 공을 들인 지원자는 본인이 어떤 내용과 경험 등을 기술했는지 구체적으로 설명이 가능하다.

이를 바탕으로 지원 회사와 연계해 본인의 강점을 어필할 수 있다. 반면 여러 회사에 같은 지원서를 제출한 지원자는 우리 회사에 대한 정보를 정확하게 파악하지 못하고 있을 확률이 높다. 그러므로 면접 때 지원자의 주요 이력 1~2개를 중점적으로 질문하여 회사가 원하는 기술과 경험을 보유하고 있는지를 재차 확인해야 한다. 이력서에 쓴 기술과 경험을 질문했을 때 명확하게 답변하지 못하는 지원자는 진실하게 이력을 작성하지 않았을 확률이 높다. 우수한 개발자들은 여러 프로젝트 경험들이 있으며 본인이 가진 기술을 명확하게 설명할 수 있다. 그렇지 못한 경우를 거르기 위해 아래와 같은 예시 질문을 준비했다.

● 면접 질문 예시

- 본인이 진행했던 프로젝트에 대해서 말해보세요: 개발자가 사용하는 프로그래밍 언어, 협업 방법, 구현한 기술 등을 전체적으로 파악 가능.
- 기존에 작성된 코드가 마음에 들지 않아 모두 지우고 새롭게 작

성해본 경험이 있는지 말해보세요: 기존 업무를 다양한 시각으로 살펴볼 수 있는 역량과 비판적 사고, 문제 해결 능력 등을 파악 가능.

- 지원자가 작성한 코드 중 가장 자신 있는 코드는 무엇인가요? 그것을 구체적으로 설명하고 이것을 더욱 좋게 개선할 수 있는 아이디어가 있는지 말해보세요: 코딩 스킬을 더욱 상세하게 파악할 수 있으며 이를 기반으로 우리 회사 업무와 연계가 가능한지, 얼마나 관심이 있는지 등을 파악 가능.
- 개발할 때 주로 활용하는 협업 툴은 어떤 것이 있는지, 코드 관리는 어떤 툴을 활용하는지 등 다양한 툴 활용 경험에 대해서 말해보세요: 협업 능력, 코딩의 좋지 않은 습관이나 팀 내에서 일할 때 소스 코드 관리를 어떻게 했는지 등을 파악 가능.
- 코딩에서 당신을 설레게 하는 부분은 무엇인지 말해보세요: 개발자가 일을 즐기며 할 수 있는 역량을 보유하고 있는지, 일에 대한 열정을 가졌는지 등을 파악 가능.
- 개발자가 아닌 다른 사람에게 본인이 만든 개발 솔루션에 대해 설명하거나 발표해본 경험을 말해보세요. 만약, 해보았다면 어떤 방식으로 이해를 시켰는지 구체적으로 말해보세요: 개발자가 아닌 다른 사람과의 의사소통 방식과 역량을 파악 가능.
- 개발자로 일하면서 상사와의 의견이 다른 경우가 많이 발생할 수 있는데 이에 대처하는 방법을 말해보세요: 동료 및 상사와 협업 방법 및 갈등 해결 능력 등을 파악 가능.

- 본인만의 협업 방식에 대해 말해보세요: 다양한 분야의 사람들과 협업하고 의사소통 방식은 어떤지 파악 가능.
- 팀의 리더로 근무해본 경험이 있는지? 본인의 리더십은 어떤 유형인지 말해보세요: 개발팀을 이끌어갈 수 있는 역량 보유 여부, 팀원들 간의 협업 및 소통 방식 등을 파악 가능.
- 탐색(트리, 정렬) 알고리즘에 대해서 종류와 특징 등을 말해보세요, 정렬 방식 중 가장 빠른 방식은 무엇인가요?: 개발자가 갖추어야 하는 기본 지식(자료구조와 알고리즘)을 파악 가능.
- 프로젝트 수행 중 어려움을 겪은 경험이 있나요? 이런 상황이 생기면 어떻게 소통할 것인지 말해보세요: 자신의 단점이나 실수 등을 인정하고 이를 극복할 수 있는 역량을 가졌는지를 파악 가능.
- 혹시 현재 회사에 대한 불만(불편함)이 있나요? 만약 있다면 이것을 해결하기 위해서 어떤 노력을 했나요, 현재 회사에서 같이 일하기 싫은 동료는 어떤 유형인가요?: 회사와 동료에 대한 불만을 표출하는 방법과 갈등 해결 역량을 파악 가능.
- 프로젝트 데드라인을 지키지 못했을 때가 있었나요? 만약 있다면 그 이유와 이를 해결한 방법에 대해서 말해보세요: 시간 관리 능력과 위기 관리 능력, 순발력 등을 파악 가능.
- 우리 회사에서 이 업무를 수행하면서 배우고 싶은 것은 무엇이며, 궁극적으로 추구하고자 하는 목표는 무엇인지 말해보세요: 회사에 대한 관심과 본인이 가진 개발 철학을 기반으로 한 성장 가능성 등을 파악 가능.

● 면접 진행 시 잊지 말아야 할 것들

면접관들이 흔히 오해하는 것이 있는데 스스로가 아주 우월한 자리에 있다는 착각이다. 면접은 면접관을 뺀내기 위한 자리가 아니다. 지원자가 면접관보다 해당 기술에 대해 더 많이 알 수도 있다. 절대 지원자를 대상으로 가르치려고 들지 말아라. 면접관은 본인의 지식을 기반으로 지원자가 최대한 많은 말을 할 수 있도록 도움을 주는 역할을 해야 한다. 그 역할을 잘 수행해야 짧은 시간 내에 회사에 필요한 답변을 끌어낼 수 있다.

질문의 수준을 한층 높일 필요가 있다. 단순하게 사는 곳, 성격의 장단점 등을 물어보는 것보다는 면접 전 지원자의 이력서를 꼼꼼하게 읽어 심층적으로 이해하는 것이 선행되어야 한다. 이렇게 이력서를 심도 있게 이해하게 되면 지원자가 가진 고민을 조금이나마 파악할 수 있다. 이를 바탕으로 다양한 문제를 회사와 함께 해결해 나갈 수 있다는 신뢰를 심어줘야 한다. 사전에 회사의 개발 문제점을 몇 가지 준비하고 이를 해결하는 방안과 지원자의 경력과의 일치도 여부를 검증하는 것도 좋은 면접 방법이다.

개발자를 대상으로 면접을 할 때는 반드시 기술 면접을 진행할 필요가 있다. 회사에 전문가가 없다면 외부 전문가를 초빙해서 개발과 관련된 실제 문제 상황, 보유 기술, 업무 등에 대해서 기술 면접을 진행해야 한다. 면접 전에는 면접관들이 모여서 분야별로 할 질문을 사전에 조율하는 것이 좋다. 기술 면접은 지원자의 경험과 현재 업무 등 개인에게 맞는 질문을 하는 것이 좋으며 단순 질문 형식이 아닌 대화 형

식으로 진행하는 것이 좋다. 만약, 지원자가 질문에 대한 대답을 어려워하면 힌트를 제공해 면접을 잘 마무리할 수 있도록 도와주는 것도 바람직한 방법이다. 이러한 기술 면접 없이 일반적인 질문들로 끝난다면 지원자는 면접이 매우 쉽다고 느낄 것이다. 그리고 회사가 보유한 기술이 실제로 구현이 가능한지, 관련 개발자들이 있는지 등에 의문점이 생길 수밖에 없다.

다음은 우리 회사의 개발 문화에 대해 어떻게 생각하는지를 파악해야 한다. 지원자가 회사 내 코드 리뷰, 에자일, 기획자, 협업 방식 등에 대해서 질문한다면 개발에 적극적으로 참여하고 싶어 하며 이에 대해 주도권을 갖고 싶어 한다는 의미로 해석할 수 있다. 이 회사에서 어떤 방식으로 일할지 고민한 지원자라면 회사의 개발 문화에 관심을 갖고 더욱 구체적으로 물을 수 있다. 반면 관심이 없고 그냥 한 번 지원해본 정도로 면접에 참석한 개발자는 이에 대해 깊이 있게 고민하지 않았을 것이다. 면접관의 구체적인 질문에도 추상적인 답변만 내놓을 확률이 높다.

회사에서도 지원자에게 큰 관심이 있다는 모습을 보여라. 지원자가 맡고 있는 업무에 관심이 많으며 이 업무가 무엇을 하는 업무인지 상세하게 물어보라. 지원자 본인이 가장 잘 아는 부분이기 때문에 자신 있게 설명할 수 있을 것이다. 그 후 천천히 질문의 수준을 확장해 나간다면 더욱 부드러운 인터뷰가 될 수 있다. 현재 하고 있는 업무와 우리 회사의 업무가 비슷하다면 이를 바탕으로 질문을 이어나가면 좋다. 이를 통해 우리 회사에 꼭 필요한 업무 경험이 있는지를 파악하기도 쉬울 것이다.

선정하기

이제 지원자에 대한 탐색이 모두 끝난 단계이다. 인터뷰를 통해 대부분의 정보를 파악했으며 우리 회사와의 일치도 여부도 어느 정도는 파악이 되었을 것이다. 이제 최종 결정만 남은 상태다. 사실 이 단계가 제일 중요하다. 우리가 뽑고자 하는 사람이 없을 수도 있고 너무 여러 명이라 선별하기가 곤란할 수도 있다. 다음에서 설명하는 가이드에 따라 최종 선정하는 과정을 거치면 좋은 개발자를 채용할 확률이 높아질 것이다.

● 이런 지원자는 뽑아야 한다

- **다양한 개발 언어를 활용할 수 있는 역량이 있는 지원자:** 새로운 기회가 왔을 때 꾸준히 학습하는 유형이다. 이러한 개발자들은 새로운 기술이나 배우고 싶은 언어가 있으면 관련 서적 및 자료들을 능동적으로 찾아서 학습해 지식과 경험을 축적한다. 이런 개발자를 채용하면 다양한 분야에서 활용할 수 있다는 장점이 있다. 이러한 역량을 바탕으로 회사에서 새로운 기회를 창출하는 데 많은 도움을 줄 수 있다.
- **좋은 소프트 스킬(Soft Skill)을 가진 지원자:** 많은 회사에서 뽑고자 하는 유형이다. 소프트 스킬(Soft Skill)은 개발 관련 역량 등 전문성을 나타내는 하드 스킬(Hard Skill)과는 반대되는 개념이다. 소프트 스킬은 의사소통, 협업 능력, 시간 관리 능력, 문제 해결 능력 등을 통칭한다. 좋은 소프트 스킬을 가진 지원자는 수용적이고

겸손한 태도, 솔직함, 높은 자존감, 일에 대한 끈기를 가진 것이 특징이다. 이런 태도를 보이는 지원자는 좋은 역량을 가진 선배 개발자(사수)를 만나면 빠르게 회사에 적응할 수 있다. 발전 속도와 실력 성장 또한 매우 빠르다. 또한, 이런 지원자는 다양한 상황 변화에 따른 대응력도 우수하여 새로운 사업 등을 추진하는 데 중요한 역할을 할 수 있다.

하드 스킬은 아주 훌륭하지만 소프트 스킬은 떨어지는 경우, 본인 분야에 대한 자부심이 넘쳐 우월감으로 변질될 수 있다. “Java를 활용하는 사람들은 같이 일하기 힘들다”, “이렇게 개발을 할 수 있는 사람은 나밖에 없다” 등의 태도를 보일 수 있다.

이런 유형의 지원자는 채용하지 않는 편이 바람직하며 앞서 언급한 소프트 스킬이 높은 지원자를 채용하는 것이 좋다.

- **끊임없이 공부하는 개발자:** 개발 분야는 끊임없이 변화한다. 새로운 기술이 계속 등장하고 기존 기술의 활용도가 점점 낮아지기도 한다. 회사에는 이러한 변화에 잘 대응할 수 있는 역량을 가진 개발자가 필요하다. 이들은 새로운 분야를 공부하는 것을 게을리하지 않는다. 이런 개발자들은 크게 두 가지 유형으로 분류할 수 있다. 첫 번째는 지식이 넓은 타입의 개발자이다. 이들은 매우 다양한 기술과 프로그래밍 언어를 이해하고 활용할 수 있다. 항상 배울 자세가 되어있어 최신 기술이 등장하는 것에 두려움을 느끼지 않는다. 두 번째는 하나의 특정 기술을 아주 깊게 이해하고 있는 개발자이다. ‘나는 천 가지 발차기를 한 사람은 무섭지 않다. 내가

제일 두려운 사람은 한 가지 발차기를 천 번 연습한 사람이다'라는 명언이 있다. 특정 기술을 수년 이상 갖고 닦은 경험은 결국 해당 분야의 전문가를 만들게 된다. 한 분야에 대해 꾸준하게 깊은 학습을 한 개발자라면 우리 회사에 꼭 필요한 인재가 될 것이다.

- **협업을 잘하는 개발자:** 제품을 개발하기 위해서 가장 중요한 건 협업 능력이다. 고품질의 제품(서비스)을 효율적으로 개발하기 위해서는 고려해야 할 다양한 요소들이 있다. 이러한 요소들이 조화를 이루는 데 가장 큰 역할을 하는 것은 바로 사람이다. 동료 간의 협업이 원활하게 이루어지지 않는다면 이러한 요소들이 조화롭게 움직일 수 없다. 실제로 개발자들은 많은 사람들과 협업을 한다. 디자이너, 기획자 등 개발 업무와 전혀 무관한 분야의 사람들과도 종종 함께 일을 한다. 본인의 분야만 잘한다고 좋은 개발자가 되지 않는다. 다양한 유형의 사람들과의 협업을 통해 시너지를 낼 수 있는 사람이 필요하다. 이러한 유형의 개발자를 채용한다면 긍정적인 결과를 보여줄 수 있을 것이다.

- **이런 지원자는 피해야 한다**

- **시간 약속을 어기는 지원자:** 아무리 다양한 경험과 활용 가능한 기술이 많더라도 피해야 할 지원자의 유형이 있다. 가장 중요한 것은 시간 약속이다. 면접 시간이 지나도록 나타나지 않는 지원자에게 인사담당자가 전화를 걸어 “어디쯤 도착하셨어요?”라고 묻는 일은 종종 일어나는 해프닝이다. 담당자와 처음으로 대면하게

되는 상견례 자리에서 이런 행동을 보이는 지원자가 회사에서 긍정적인 방향으로 성장할 수 있을까? 정답은 '아니오'이다. 사람은 쉽게 바뀌지 않는다. 항상 그런 패턴으로 삶을 살아왔기 때문에 면접 시간에도 늦는 것이다. 면접 시간에 늦었다면 아무리 실력이 좋더라도 채용을 하지 않는 것이 좋다.

- **소프트 스킬이 불량한 지원자:** 유능한 개발자라도 소프트 스킬이 불량하면 뽑지 않는 것이 좋다. 이러한 개발자는 협업이 중요한 프로젝트에서 불편한 존재로 전락하기 마련이다. 오픈 소스 시대에 혼자 모든 것을 개발하는 개발자는 없다. 그만큼 협업이 매우 중요한 시대다. 소프트 스킬이 낮은 개발자는 함께 일하는 동료들의 마음을 불편하게 만들어 개발의 속도와 퀄리티를 떨어뜨리는 원인으로 작용할 확률이 높다. 혹시나 이런 개발자를 채용해야 한다면 가급적 이른 시일 내에 좋은 태도를 가진 개발자로 변화시켜야 한다. 이것이 말처럼 쉬운 것은 아니지만 강한 동기부여와 멘토링 등을 통해 어느 정도는 변화시킬 수 있을 것이다. 잘 변화가 된다면 그만큼 회사에 대한 신뢰와 충성도가 높아졌다고 볼 수 있다. 오히려 다른 개발자들보다 장기 근속할 확률이 높아지는 결과를 가져오기도 한다.
- **불만이 많은 개발자:** 이는 긍정적인 마인드와 연계할 수 있다. 실제로 개발자 중 이러한 유형의 개발자도 종종 있기 마련이다. 면접을 통해 이를 확인하기는 쉽지 않지만 면접 시 많은 대화를 통해 이러한 성향을 조금이나마 파악하는 것이 좋다. 앞서 면접 질문

으로 소개한 현 회사에 대한 불만족 요인, 동료에 대한 불만, 이직 후 우리 회사에서 어떤 부분을 개선했으면 좋겠는지 등의 질문을 통해 파악하는 것이 좋다.

- **본인 자랑만 하는 개발자:** 자신이 했던 개발 경력을 자랑하듯이 말하는 개발자들이 있다. 이러한 유형은 대개 40대 이상의 개발자 중에서 많이 나타난다. 자신의 경험을 상세하게 이야기하는 건 좋다. 하지만 부담스러울 정도로 지나치게 이야기하는 사람이 있다. 이런 개발은 나만 가능한 일이며 본인을 뽑으려면 뽑고 아니면 말라는 태도를 보일 수 있다. 소개와 자랑은 엄연히 다르다. 개발자는 경력별로 해야 할 일들이 어느 정도 나누어져 있다. 20~30대 개발자들은 다양한 개발 업무를 접하고 현장에서 많은 경험을 쌓아야 하는 시기이고, 40대 이상의 개발자는 세부 구현 기술보다는 전체 도메인에 대한 이해도가 높아야 한다. 자신의 능력을 자랑하는 40대 이상의 개발자라면 전체 도메인에 대한 이해도와 팀의 구성원을 이끌 수 있을지를 명확하게 확인해야 한다. 그렇지 않으면 고집이 센 개발자 한 명을 뽑아서 앓혀 놓는 것과 다를 바 없다.

● 개발자의 학벌은 실력을 대변하지 않는다

합격자 선정의 우선순위를 학벌에 두면 어떤 결과가 나올까? 물론 좋은 학벌을 가진 개발자들은 많다. 하지만 학벌은 떨어지지만 훌륭한 역량과 경험을 가진 개발자들의 수는 훨씬 많다. 동일한 역량과 경험, 인성

을 갖고 있다는 전제에서는 학별로 판단하는 일이 도움이 될 수 있다. 그러나 학별을 최종 판단 기준으로 삼으면 안 된다.

자격증도 이와 유사하다. 한국은 유독 자격증을 중시하는 문화가 있다. 관련 자격증이 있다고 해서 실제로 개발을 잘할 수 있을까? 자격증 수십 개를 보유한 사람이 다양한 개발 역량을 가지고 있을까? 자격증을 갖고 있다면 업무에 도움은 되겠지만 자격증이 합격의 기준이 되어서는 안 된다. 불과 며칠 전에 공부한 내용도 다시 보지 않으면 잊어버리기 마련이다. 수년 전에 취득한 자격증이 개발자의 실력을 대신할 수 있을 것으로 생각하지 않기를 바란다. 개발자 채용 시 전공자와 비전공자를 구분하여 채용하는 것은 어떨까? 전공자라도 개발이 적성에 맞지 않는 사람이 있는가 하면 비전공자라도 개발이 적성에 맞는 사람이 있다. 최근에는 유튜브 등으로 소프트웨어 개발 기술을 배울 수 있는 곳이 넘쳐나고 있다. 비전공자라도 개발이 적성에 맞다면 충분히 좋은 성과를 낼 수 있는 환경인 것이다. 앞으로는 전공과 비전공의 경계가 모호해질 수 있다. 전공이 중요한 게 아니라 적성에 맞는지가 중요한 시대가 되었다.

개발자 채용 시에는 학별, 자격증과 같은 유형적인 요소가 아닌 실제로 얼마나 다양한 ‘경험’과 ‘실력’이 있는지를 파악하는 것이 중요하다. 또한, 채용 분야와 ‘전공’의 일치도보다는 ‘적성’의 일치도를 기준으로 채용해야 한다. 본인 적성에 맞아 일을 즐기는 사람을 이길 자는 없다. 최종 합격자 선정을 위한 중요한 기준이 학별, 자격증, 전공이라면 우수한 개발자 채용으로 이어지기 힘들다는 사실을 잊지 말아라.

● 레퍼런스 체크를 꼭 해라

기업 인사담당자를 대상으로 경력직 채용 시 이전 직장에 평판 조회(레퍼런스 체크)를 해본 경험이 있느냐는 질문에 약 70% 정도가 있다는 응답이 나왔다. 그렇다면 이러한 레퍼런스 체크를 통해 어떤 것을 파악할 수 있을까? 사람인(<https://www.saramin.co.kr/>)에서 조사한 바에 따르면 가장 알고 싶은 것은 지원자의 인성과 성격(77.6%)이었고, 다음으로는 전 직장 퇴직 사유(57.8%)와 업무 능력(50%) 순이었다. 다시 말하면 채용 시 가장 중요하게 보는 것은 지원자의 인성과 성격이라는 말과 다를 것이 없다. 면접 당시에는 웃는 모습으로 긍정적인 태도를 보인 지원자가 레퍼런스 체크 결과 동료들이 싫어하는 유형의 직원일 수도 있다. 지원자가 입사 전까지는 회사 분위기를 모르는 것과 같이 회사도 같이 근무하기 전까지는 지원자에 대해서 잘 알지 못한다. 따라서 레퍼런스 체크를 통해 지원자의 인성과 업무 역량 등을 파악할 수 있다면 많은 도움이 된다. 하지만 동료의 말을 전적으로 신뢰하는 것도 좋지 않다.

앞에서 언급한 조사에 따르면 ‘레퍼런스 체크가 실제 채용에 미치는 영향은 어느 정도인가’라는 질문에 과반수 이상이 ‘참고만 한다(55.4%)’고 응답했다. ‘평가 점수에 반영한다’는 응답은 29.4%로 대부분 참고하는 수준에 그치는 것으로 나타났다. 우리나라 문화 특성상 남에 대해서 좋지 않은 말을 외부에 하는 것을 꺼리는 경향이 있다. 때문에 레퍼런스 체크 전화가 오면 ‘그냥 괜찮은 사람이다’ 정도로 마무리하는 경우가 많다. 그러므로 레퍼런스 체크는 하지만 참고만 한다는

의견이 많은 것으로 생각된다. 그렇지만 최종 채용 전에는 꼭 진행하는 것이 좋다. 평판이 좋지 않은 지원자를 걸러내는 마지막 기회이기 때문이다.

Offer

최종 채용 대상자가 선정되었다면 그들에게 최종 오퍼(Offer)를 보내야 한다. 그런데 유능한 개발자들은 이미 여러 회사로부터 오퍼를 받고 있을 확률이 높다. 실제로 합격한 지원자에게 오퍼를 보내면 반 이상의 지원자들이 거절한다. 해당 지원자들은 이미 차선의 선택지가 많음을 알아야 한다. 반드시 채용하고 싶은 인재라면 업무 포지션, 근무하게 될 팀 등을 적극적으로 마케팅해야 한다. 해당 지원자에 대한 분석을 통해 지원자가 우선시하는 요구들을 이미 알고 있을 것이다. 이에 맞춰서 우리 회사에서 근무하게 되었을 때의 좋은 점들을 강하게 어필하자.

오퍼의 내용

- **연봉 및 보상 체계:** 대부분의 지원자가 제일 중요하게 생각하는 요소이다. 최소한 지원자의 현 직장의 연봉 수준보다는 높게 책정해야 이직의 메리트를 줄 수 있다. 우선 회사에서 협상 가능한 범위를 책정하고 그 범위 내에서 조율을 시도한다. 다른 회사에서 제공하지 않는 특별 성과급, 입사 축하금, 복지 혜택 등이 포함된 보

상 패키지를 제공하는 것도 이직을 위한 큰 요소가 될 수 있다.

- **담당하게 될 역할과 책임, 상위 보고자:** 일하게 될 팀에서 해야 할 업무와 책임에 대해 구체적으로 설명하라. 개발자 특성상 수평적인 기업 문화를 선호하는 경우가 많다. 만약 상위 보고자가 있다면 어떤 성향이며 어느 상황에서 보고가 이루어져야 하는지 등을 말해주는 것이 좋다. 그래야 우리 회사에 들어온 지 얼마 되지 않아 다른 회사로 이직하는 것을 방지할 수 있다.
- **다양한 복지 혜택:** 채용 공고에 이미 간략한 내용은 기재되어 있으나 더욱 구체적인 내용을 언급할 필요가 있다. 피부에 와닿지 않는 여러 개의 복지보다는 바로 체감할 수 있는 ‘개인 사무실 제공’, ‘식사 비용 지원’, ‘주 1회 재택 근무’, ‘자율 출퇴근’ 등의 혜택을 강조해 우리 회사를 적극적으로 어필할 수 있다.
- **우리 회사의 제도(문화) 소개:** 우리 회사에서 추구하는 가치와 문화가 있다면 적극적으로 알리는 것이 좋다. 또한, 지속적인 소통과 수용적인 분위기의 회사라는 것을 알려주면 지원자가 우리 회사를 선택하는 데 많은 도움을 줄 수 있다.
- **스톡옵션 제공 기회:** 회사 규모가 크지 않다면 스톡옵션을 제공하는 것도 좋은 방법이다. 스톡옵션을 받으면 회사의 성장은 개발자가 소유한 스톡옵션 가치의 상승으로 이어지므로 소속감이 훨씬 높아지는 장점이 있다. 또한, 새로운 업무가 주어지더라도 이에 맞는 보상이 올 확률이 높아지는 것이므로 더욱 긍정적인 자세로 일을 수행할 것이다.

- **수습 기간:** 수습 기간이 종료된 후 연봉 협상을 다시 하는 것도 좋은 방법이다. 수습 기간 중에 함께 일을 해보면 해당 개발자의 역량을 충분히 파악할 수 있다. 처음 기대했던 것보다 개발(업무) 역량이 뛰어나다면 수습 기간 이후에 다시 연봉 협상이 가능하다는 언질을 줘야 한다.

오퍼를 보낼 때 주의사항

위에서 언급한 내용을 바탕으로 오퍼를 보낼 때 유의해야 할 사항이 있다. 이를 간과하여 우수한 개발자를 놓치는 우를 범하지 않기를 바란다.

● 빠른 속도로 진행해라

앞에서 언급한 것처럼 우수한 개발자는 많은 옵션을 가지고 있다. 우리 회사 말고도 갈 수 있는 회사들이 많다는 얘기가. 따라서 이 사람을 꼭 채용해야겠다는 생각이 들면 적극적으로 다가가는 자세가 필요하다. 개발자 채용은 타 기업과의 경쟁이다. 다른 기업에게 지원자를 뺏기고 싶지 않다면 신속하게 움직여야 한다. 우물쭈물하다 우수한 개발자가 경쟁 회사로 가면 그 회사가 더욱 성장하도록 도와주는 꼴이 되어버린다.

최종 채용 대상자가 결정되었다면 시간을 지체하지 말고 바로 오퍼를 보내야 한다. 오퍼를 보낼 때 앞에서 언급한 내용을 구체적으로 보내고 우선으로 상세한 설명을 해줘야 한다. 자칫하면 다 된 밥을 다시

해야 하는 상황이 올 수 있다.

금요일에 이력서를 받아 토, 일, 월 3일 동안 면접과 채용까지 진행한 사례도 있다. 이것은 특정 사례를 예시로 든 것이지만 그만큼 신속하게 진행해야 한다는 것을 말해주는 것이다.

채용 대상자에 대한 빠른 피드백도 중요하다. 최종 채용 대상자가 오퍼를 받은 후 연봉과 복지 등의 질문을 하면 인사담당자는 그에 대한 즉각적인 피드백을 보내야 한다. 바쁘다는 핑계로 피드백이 늦으면 지원자는 다른 옵션을 고민할 확률이 높다.

● 정성을 들여야 한다

채용하기로 한 지원자에게 오퍼를 보낼 때는 최대한 정성을 들여야 한다. 일반적으로 전화를 통해 합격 여부를 통지하고 공식적인 채용 절차 및 일정 등에 대해서는 이메일로 다시 한번 송부한다. 이때 설명이 부족한 내용은 언제든지 인사담당자에게 전화로 문의할 것을 명시해야 한다. 또한, 연봉이나 복지 등이 마음에 들지 않아 고민할 때에는 해당 지원자에게 해줄 수 있는 모든 옵션을 보여줘야 한다. 이런 내용을 전달할 때는 될 수 있으면 전화로 정중하게 안내하고 빠르고 즉각적인 대응을 하는 것이 좋다.

- ✔ 개발자 채용 자체가 목적이 되어서는 안 되며 뽑으려는 목적이 명확해야 한다.
- ✔ 개발자 채용을 통하여 회사의 이익이 발생해야 한다.
- ✔ 명확한 기준과 목적이 없는 상태에서 채용을 하면 후회한다.
- ✔ 뽑을 수 있다면 한 번에 최소 2명 이상의 개발자를 채용하는 것이 좋다
- ✔ 회사의 규모 및 상황에 따라 개발자 채용 목적을 달리 설정해야 한다.
- ✔ 개발자 채용을 위해 서비스 플랫폼 형태, 필요 기술, 개발 환경, 프로그래밍 언어 등을 사전에 명확하게 설정하는 것이 좋다.
- ✔ 우리 회사에 필요한 인력의 등급, 소요 예산 등을 고려하여 필요한 개발자의 등급이 어느 정도인지 파악하는 것이 좋다.
- ✔ 개발자 인소싱은 품질 관리, 빠른 의사소통 등의 장점이 있으나 시간과 비용이 많이 든다는 단점이 있다.
- ✔ 개발자 아웃소싱은 시간과 비용을 절약할 수 있다는 장점이 있으나 품질 관리 및 의사소통의 어려움 등의 단점이 있다.
- ✔ 임원진(경영진)이 개발 및 IT 분야를 알면 좋은 개발자를 뽑을 수 있는 기반이 된다.
- ✔ 채용담당자(인사팀)는 외부의 다양한 요인을 분석해 효율적인 채용 전략을 수립해야 한다. 도움이 필요할 경우 개발자들과 함께 일을 하는 것이 좋다.
- ✔ CTO/개발팀장/시니어 개발자는 지원자에 대한 철저한 기술 면접 등으로 실력을 파악하는 데 힘써야 한다.
- ✔ 개발자들은 생각보다 적극적인 구직활동을 하지 않으니 우리 회사의 채용 사실을 적극적으로 알려야 한다.
- ✔ 채용 실행은 Sourcing-Screening-Interview-Offer의 4단계로 구분된다.

- ✔ 직무 기술서(JD)는 해당 부서의 개발자가 같이 참여하여 명확하게 작성한다. 실제 담당 업무, 필요 기술, 연봉 및 복지 등에 대한 명확한 설명이 필요하다.
- ✔ 좋은 개발자는 내부 직원이 더 잘 알고 있는 경우가 많으므로 직원 추천제도를 적극적으로 활용한다.
- ✔ 개발자 채용에 전문성이 높은 헤드헌팅(서치펌)을 적극적으로 활용하여 효율적인 채용을 진행하는 것도 좋은 방법이다.
- ✔ 이력서는 세심하게 검토하여 지원자의 기초적인 정보를 충분히 파악하고 채용에 임하는 것이 좋다.
- ✔ 코드 리뷰 혹은 코딩 테스트 등의 과정을 통해 개발자가 얼마나 전문성을 갖추고 있는지 검증하는 것이 필요하다.
- ✔ 코딩 테스트는 회사에서 실제로 발생하는 문제를 중심으로 역량을 파악하는 것이 좋다.
- ✔ 고급 개발자 채용 시에는 코딩 테스트로 검증하기 어려우므로 깃(Git) 저장소 등의 코드 퀄리티 등을 종합 평가하는 것이 좋다.
- ✔ 인터뷰 전 지원자에 대한 꼼꼼한 검토와 지향하는 목적 등을 파악하는 것이 좋다.
- ✔ 전화(화상) 인터뷰를 진행할 경우 구체적인 기준을 잡고 이에 따라서 진행하는 것이 좋다.
- ✔ 인터뷰 시 지원자와 우리 회사가 얼마나 궁합이 잘 맞는지, 어떤 경험과 기술이 있는지 등을 다양한 질문을 통해 심층적으로 파악하는 것이 좋다.
- ✔ 면접관은 지원자가 대답을 잘할 수 있도록 도움을 주는 역할을 해야 한다. 피상적인 질문이 아닌 지원자 개인에 맞는 질문을 준비하는 것이 좋다.
- ✔ 개발자 면접 시에는 반드시 기술 면접을 진행해야 하며 대화 형식으로 자연스럽게 그들이 가진 기술을 파악하는 것이 좋다.

- ✔ 지원자 중 다양한 언어 활용이 가능한 개발자, 좋은 소프트 스킬을 가진 개발자, 끊임없이 공부하는 개발자, 협업에 능한 개발자 등은 반드시 뽑는 편이 좋다.
- ✔ 지원자 중 시간 약속을 어기는 개발자, 불만이 많은 개발자, 본인 자랑만 하는 개발자 등의 유형은 뽑지 않는 편이 좋다.
- ✔ 개발자의 학벌, 전공, 자격증은 실력을 대변하지 않으므로 최종 합격의 기준으로 삼지 않는 것이 좋다.
- ✔ 개발자들에 대한 평판 조회(레퍼런스 체크)를 꼭 해볼 필요가 있다. 평판이 좋지 않은 지원자는 걸러내는 것이 좋다.
- ✔ 지원자에게 오피를 보낼 때는 연봉 및 복지, 회사의 제도 및 문화, 담당 업무, 스택 옵션 등 최대한 자세한 내용을 적어서 보내는 것이 좋다.
- ✔ 오피를 보낼 때 최대한 빠른 속도로 진행할 필요가 있으며 정성을 담아 보내는 것이 좋다.



4.

관리보다는 케어하라

- 01 | 떠나기 싫은 회사 만들기
- 02 | 개발자들과 소통하는 법
- 03 | 팀의 성장은 회사의 성장이다



01

떠나기 싫은 회사 만들기

엄청난 노력을 해서 뽑은 개발자가 입사 3개월 만에 다른 회사로 이직해버리는 일이 발생했다. “진짜 어렵게 뽑은 개발자인데...”라고 후회해도 이미 늦었다. 옛말에 “잡은 물고기한테는 먹이를 주지 않는다”라는 말이 있다. 이미 뽑았으니, 우리 회사의 전유물로 생각하고 정성껏 보살펴주지 않으면 개발자들은 곧 떠날 준비를 할 것이다. 그렇다면 공들여서 뽑은 개발자가 떠나지 않게 하려면 어떻게 해야 할까? 개발자가 쉽게 떠나는 이유는 무엇일까? 이번 장에서는 개발자가 오랫동안 일하고 싶은 회사를 만드는 방법을 소개한다.

시작에 앞서 강조하고 싶은 것이 있다. 개발자들을 케어하기 위한 방법 중에는 회사에서 소화하기 힘든 것들도 있을 것이다. 여기에서 소개하는 모든 방법을 회사가 적용하기는 쉽지 않은 일이다. 가장 중요한 것은 ‘개발자들을 진심으로 대하는 마음’이다. 가족처럼 아끼고 진심으로 대한다면 개발자들은 우리 회사와 오랫동안 함께할 수 있을 것이다.

개발자에게 집중하는 것은 당신 몫이다

일에서 의미를 찾도록 도와주는 역할을 하라

세계적인 건축물로 알려진 런던의 성 바오로 성당을 설계한 크리스토퍼 렌의 유명한 일화를 소개한다. 그는 성당 건축 현장에서 일하는 노동자에게 지금 무엇을 하고 있는지 물었다. 대부분의 노동자는 “벽돌을 나르고 있습니다” 또는 “일을 하고 있습니다” 등의 성의 없는 대답을 했다. 그들은 돈을 벌기 위해서 어쩔 수 없이 일하는 사람들이었다. 그런데 한 노동자의 행동과 말은 이들과 달랐다. 그는 흥얼거리며 일을 즐기는 듯했으며, “나는 전능한 신을 위한 장엄한 성당을 건축하고 있습니다”라고 대답했다. 이 노동자는 힘들고 고된 일을 즐겼고 하는 일에 긍정적인 ‘의미’를 부여했다. 이런 유형의 사람들은 본인이 하는 일에서 ‘의미’를 발견할 때 더욱 열중할 수 있다. 반대로 ‘의미’를 발견할 수 없다면 일의 능률은 자연스레 떨어질 수밖에 없다. ‘노력하는 자는 즐기는 자를 이길 수 없다’라는 말이 있지 않은가. 개발자 세계에서 이런 원칙이 더욱 공고해진다. 개발자 대부분은 새로운 것을 배우는 것에 대한 흥미와 자신이 하는 일에 대한 엄청난 자부심이 있다. 이들은 본인의 일에서 ‘의미’를 발견하면 어떤 일이 주어지더라도 재미있게 일한다. 그러므로 우리는 그들이 하는 일에서 ‘일의 의미’를 찾을 수 있도록, 그리고 즐겁게 일할 수 있도록 도와줘야 한다.

개발자가 만족할 수 있는 환경을 만들어라

그럼 우리는 개발자들이 스스로 의미를 찾기를 기다려야 하는가? 아니면 그들이 의미를 찾을 수 있도록 도와주어야 하는가? 우선 개발자들이 회사에서 찾을 수 있는 의미란 무엇인가를 고민해야 한다. 여러분이 개발자라면 어떤 것으로 회사 일에 의미를 둘 수 있을까? 대부분은 앞서 소개한 노동자와 비슷한 대답을 할 것이다. ‘돈을 많이 벌기 위해서’, ‘이름만 대면 알만한 좋은 회사에 다니기 위해서’, ‘최고의 복지제도를 누리기 위해서’, ‘자신의 커리어를 위해서’ 등 유형적 가치를 중시하는 답변을 할 수 있을 것이다. 그렇다면 이러한 조건을 대부분 갖춘 곳은 어디인가? 사실 스타트업 등에서 위의 조건을 모두 맞춰 주는 현실적으로 쉽지 않다. 대기업은 되어야 위의 조건 대부분을 만족시켜줄 수 있을 것이다. 만약 ‘조건’을 중시한다면 대기업에 취직하는 것이 본인이 추구하는 삶의 가치와 많은 부분에서 일치할 수 있다. 그리고 이곳에 가서도 ‘일의 의미’를 충분히 찾을 수 있다.

하지만 이 글을 읽고 있는 독자의 대부분은 중소·중견 기업, 스타트업 등의 작은 규모의 기업을 운영하거나 종사하고 있을 것이다.

대기업과 비교해 상대적으로 열세인 우리 회사에서 어떻게 유형적 가치에 해당하는 의미를 만들어줄 수 있을까?

냉정하게 말하면 복지와 연봉, 네임 밸류 등 어느 하나 제대로 해줄 수 없는 경우가 많다. 그럼 우리 회사는 좋은 개발자를 뽑지 못하고 바라만 보아야 하나? 아니다. 우리도 할 수 있다. 개발자 입장에서 우리 회사에 올 수 있는 다른 ‘일의 의미’를 찾을 수 있도록 도움을 주어야

한다.

우리가 내세울 것은 무엇인가? 가장 먼저 회사의 가치와 비전을 알릴 필요가 있다. 우리 회사에 입사하게 되면 새로운 제품을 만들어 볼 수 있고, 한 번도 경험해보지 못한 일을 최고의 개발자 팀 혹은 CEO와 함께할 수 있다. 개발자 본인이 만든 제품이 소위 말해 ‘대박’이 나서 회사의 중역으로 성장할 수 있다는 기대감을 심어주면 된다.

실제로 많은 개발자들이 이런 성장 가능성에 ‘일의 의미’를 부여하고 대기업에서 스타트업으로 이직하는 경우가 늘어나고 있다.

개발자들에게 ‘일의 의미’를 찾아주는 것은 CEO와 우리 회사의 몫이다. 개발자들이 그 의미를 느낄 수 있도록 조직 문화, 가치관, 소통 방법을 잘 만들어 놓는 준비가 필요하다. 이렇게 ‘일의 의미’를 부여할 수 있는 환경을 만들어 놓으면 개발자는 우리 회사의 매력에 빠져서 오랫동안 함께할 수 있는 직원이 될 것이다.

그저 따라하지 말아라

세상에는 정말 많은 기업들이 있다. 네이버, 카카오 등의 IT 플랫폼 분야의 기업이 있고 엔씨소프트나 넥슨 같은 게임 분야의 기업도 있다. 포스코나 현대차와 같은 제조 분야의 기업도 있다. 다양한 기업만큼이나 이들이 하는 일도 다양하다. 그렇다면 카카오와 현대차는 같은 기업 문화를 가지고 있을까? 기업 이름만 들어도 아닐 것이라는 생각이 들 것이다.

어떤 제품을 생산하고 서비스하는지, 그리고 어떤 유형의 직원들을

채용하는지에 따라서 기업 문화는 천차만별이다. 카카오와 네이버 같은 비슷한 업종의 기업이라도 CEO와 구성원에 따라서 그 문화는 전혀 다르게 형성된다.

국내뿐만 아니라 해외의 내로라하는 기업도 다양한 기업 문화를 갖고 있다. 이들이 만든 기업 문화 덕분에 그 회사는 매력적인 회사로 발전할 수 있었다. 그렇다면 다른 기업의 좋은 문화만을 골라서 우리 회사에 적용하면 성공적인 기업 문화 형성이 될까? 아니면 최근 유행하는 재택 근무, 무제한 휴가 등 개발자들을 유혹할 수 있는 다양한 제도들을 죄다 갖다 적용하면 좋은 문화가 형성될까?

한 가지 예를 들어보자.

배우 정우성이 B사의 명품 검정 코드와 파란색 셔츠를 입고 한 시상식에 나왔다. 당신은 이 옷들이 너무 좋아 보여서 모두 구매를 했다. 그리고 정우성이 입은 그대로 입어보았다. 하지만, 뭔가 이상하다. 분명 정우성이 입었을 때는 멋있었는데 왜 나한테는 잘 어울리지 않을까?

이유는 여러분도 알고 있겠지만 간단하다. 우리는 각자 키, 얼굴형, 체형 등이 다르므로 이에 맞는 옷을 입어야 한다. 다른 사람이 잘 어울리는 옷이라도 나에게서는 어색할 수 있다. 기업 문화도 이와 같다. 비슷한 업종이라도 기업 문화는 각양각색이다. 우수한 사례를 모방하는 것도 중요하지만, 무조건 좋은 것만 가져온다고 해서 우리 회사에 다 잘 맞으라는 보장은 없다.

CEO는 이런 기업 문화를 만드는 핵심적인 역할을 해야 한다. 잘된

사례를 참고하되 우리 회사의 업무 환경, 개발자들의 특성 등 다양한 요소들을 고려하여 이들이 '일의 의미'를 찾으며 근무할 수 있는 여건과 환경을 만들어야 한다. 다시 한번 강조하지만 무작정 따라 하지 말라. 회사 구성원들이 상호 협력하여 우리 회사만의 문화를 만들어가는 것이 가장 바람직하다. 개발자들이 일에 집중하고 즐겁게 일을 할 수 있도록 해주어야 하며 이런 환경을 만드는 건 회사의 몫이다.



동기는 개발자를 춤추게 한다

돈은 동기 유발에 큰 영향을 주지 않는다

어떤 동기를 부여해야 개발자들이 만족할 수 있을까? 연봉을 많이 주면 당연히 개발자들은 좋아한다. 하지만 돈이 곧 동기로 연결되지는 않는다. 깃허브(GitHub)에서는 수많은 개발자들이 본인이 만든 코드를 오픈 소스로 공개한다. 본인이 수많은 노력을 해서 만든 코드를 다른 사람이 활용할 수 있도록 무료로 공개하는 것이다. 이들은 돈을 바라 고 이런 작업을 하는 것일까? 오히려 돈을 주고 이러한 개발을 맡겼다면 퀄리티가 더 낮을 수 있다.

만약 여러분이 게임을 좋아한다고 하자. 그런데 내가 큰 흥미를 갖지 못하는 게임의 한 유저가 “최고 레벨까지 키워주면 돈을 많이 주겠소”라고 제안했다. 여러분은 돈을 벌기 위해서 이 재미없는 게임을 할 수도 있다. 그러나 돈이라는 동기가 없다면 이 게임은 쳐다보지도 않았을 것이다. 반대로 기다리고 기다리던 정말 재미있는 신작 게임이 나왔다면? 그럼 여러분은 이 게임을 밤새워서 해도 지루하지 않을 것이다. 대부분 한두 번쯤은 자기가 좋아하는 게임이나 일에 몰두하여 시간 가는 줄 몰랐던 경험이 있다.

개발자들도 마찬가지다. 이들이 개발을 즐길 수 있도록 동기를 부여해야 한다. 회사가 시켜서 하는 개발이 아닌, 동기에 이끌려서 몰두하는 개발이 되어야 한다. 이들은 본인이 즐기며 하는 개발을 좋아하는 동시에 본인의 역량을 다른 사람들이 알아주기를 바란다. 개발자들의 이런

성향을 잘 이해한다면 이들에게 동기를 부여하기가 더욱 쉬울 것이다.

개발자들은 이런 것을 좋아한다

개발자들의 동기는 다른 직원들과 조금 다르다. 개발자의 동기를 자극하는 것은 무엇인가? 개발자들은 본인이 하는 개발이 기업뿐만 아니라 국가(전 세계) 차원의 변화를 가져오기를 바란다. 앞서 깃허브 사례를 언급한 것처럼 본인이 즐거워서 개발하고, 그 결과물을 여러 사람들이 활용해주는 것을 좋아한다.

또한, 이들은 개발을 통해 새로운 것을 꾸준히 배우고 성장해 나가는 것을 좋아한다. 난이도가 낮은 개발 업무를 계속하는 것을 좋아하는 개발자는 거의 없다.

개발자 본인이 평소에 관심이 있었거나 흥미를 느끼는 분야여야 동기 유발에 좋다. 무언가를 배우고 성장하는 느낌을 받을 수 있도록 도움을 주어야 한다.

개발 중인 서비스나 기술이 미래에 유용하게 쓰일 수 있다는 가치를 심어주는 것도 좋다. 현재는 그다지 유망하지 않지만 앞으로 수년 내에 가치를 발휘할 수 있는 서비스라면 더욱 좋다. 개발자는 이런 서비스를 개발할 때 희열을 느낀다. 내가 만드는 기술이 미래의 핵심 기술이 될 것이라는 기대는 곧 회사의 발전으로 이어진다. 당연히 개발자 개인의 성장과도 연결될 수 있다.

개발자가 하는 개발 업무에 대해서 항상 인정해주고 칭찬하는 마음으로 접근해야 한다. 말로만 하는 칭찬이 아닌, 마음에서 우러나오는

격려와 인정이 필요하다. 개발자가 없다면 우리 회사가 돌아갈 수 없으므로 소중한 존재라는 것을 계속 인식시켜 주어야 한다. 이렇게 되면 회사에서 본인의 역할에 대해 다시 정립할 수 있다. 또한, 더욱 열심히 일할 수 있는 동기를 유발할 수도 있다.

개발자가 일하는 업무 환경도 신경 써야 할 부분이다. 활용하는 장비 중 불편한 것은 없는지, 업무에 방해되는 것은 없는지 등을 확인하여 개선하도록 해야 한다. CEO가 주기적으로 여러 개발자들의 고충을 듣는 자리를 마련하는 것도 좋다. 프로그램이나 장비, 도서 등 회사에서 지원해야 할 것들은 아끼지 말고 지원해야 한다. 또한, 재택 근무나 휴가가 필요하다고 하면 눈치 보지 않고 편하게 이용할 수 있도록 배려해야 한다. 개발자들은 회사가 지원한 것 이상의 결과로 보답할 것이다.

개발자도 멘토링이 필요하다

멘토링의 중요성

갓 입사한 신입 개발자들은 많은 것들을 배워야 한다. 회사의 문화, 분위기, 업무 등 하나부터 열까지 전혀 모르기 때문에 제대로 알려줄 사람이 필요하다. 경력 개발자가 이직을 한 경우도 마찬가지로의 도움이 필요하다. 그래서 많은 회사에서는 멘토링 프로그램을 운영한다. 근무 경험이 많은 멘토와 새로 들어온 멘티를 연결함으로써 회사의 적응 기

간을 단축하고 업무 효율성과 만족도를 높이는 역할을 한다.

멘토링을 통해 멘티만 성장하는 것이 아니라 멘토도 성장할 수 있다. 멘토의 경우 신입 개발자에게 관련 업무를 알려주는 것과 본인이 하는 일은 전혀 다르다. 위에서 시키는 일을 할 때는 그려지지 않았던 그림을 그려야만 멘티에게 업무에 대해 상세히 알려줄 수 있다. 이처럼 멘토링 프로그램은 멘토와 멘티, 그리고 회사 모두가 성장할 수 있는 계기를 마련해준다.

신입 개발자가 입사하여 맞는 가장 큰 행운은 높은 연봉도, 보너스도 아니다. 좋은 선배 개발자(멘토)를 만나는 것이다. 좋은 멘토는 참여할 프로젝트와 해야 할 역할을 명확히 안내한다. 신입 개발자가 성장할 수 있는 부분은 무엇인지도 알려줄 수 있다. 개발 중에 어렵거나 막히는 부분이 있으면 언제든지 물어볼 수 있는 선생님 역할도 수행한다. 멘토는 멘티가 암흑 같은 회사 생활을 시작할 때 길을 밝히는 등대의 역할을 해줄 수 있다.

만약 신입 개발자가 입사했는데 멘토가 없다면 어떨까? 이들은 회사 적응에 큰 어려움을 느낄 것이다. 개발하다 막히는 부분이 있으면 '노가다'로 해결할 수밖에 없을 것이다. 선배의 입장에서 보면 별것 아닌 문제가 신입 개발자들에게는 큰 시간 낭비와 비효율로 다가올 수 있다. 멘토링 프로그램의 중요성은 백번을 강조해도 지나치지 않다. 멘토링 프로그램은 개인의 성장뿐만 아니라 회사의 발전에도 큰 영향을 줄 수 있다. 그리고 어렵게 뽑은 개발자가 몇 개월 만에 회사를 그만두고 나가버리는 확률을 낮출 수 있는 좋은 제도이다.

멘토 되기

좋은 멘토가 되기 위해서는 어떻게 해야 할까? 좋은 멘토가 된다는 건 생각보다 어려운 일이다. 멘토는 자기가 맡은 업무에 멘토라는 새로운 업무가 추가된 것이다. 즉, 멘토링을 위한 추가 시간을 할애해야 하는데 이게 말처럼 쉽지 않다. 회사에서 이를 위한 배려가 없을 확률이 높기 때문이다. 만약 멘토가 멘토링을 위한 시간을 최소화하고 본인 업무에만 매달린다면 어떻게 될까? 멘티는 이러한 상황을 눈치채고 소극적으로 행동할 것이다. 이런 상황에서 멘토링의 질은 떨어질 수밖에 없다.

멘토링은 사람이 하는 것이기 때문에 질적 관리가 중요하다. 자신의 업무가 바쁘다고 시간 때우기 식으로 행동하면 신입 개발자들에게 좋지 않은 경험을 줄 수 있다. 결과적으로 어렵게 뽑은 개발자를 놓칠 수도 있다.

멘토의 말과 행동에 주의를 기울이자

신입 개발자인 멘티는 일을 배우기 위해서 부단한 노력을 할 것이다. 멘토에게 잘 보이기 위해서 긴장할 수도 있다. 멘티가 잘 적응할 수 있도록 도와주는 것이 멘토의 역할이다. 멘티가 개발 혹은 업무 관련 질문을 한다면 귀담아서 들을 필요가 있다. 이 질문은 내가 처음 개발자로 회사에 들어왔을 때 느꼈었던 의문점일 확률이 높다. 멘티의 말과 행동을 최대한 이해하는 노력도 필요하다. 신입 개발자가 하는 행동을 자세히 관찰하면 말로 할 수 없는 것들이 표현되기도 한다.

이렇듯 멘토는 멘티가 무엇을 원하는지, 어떤 부분에서 어려움을 겪고 있는지를 파악하고 적시에 도와줄 준비가 되어있어야 한다. 좋은 멘토가 되었다면 멘티는 나의 업무와 개인적인 성장에도 도움을 줄 것이다. 멘토는 선배로서 이미 겪었던 경험들을 생생하게 공유해 줄 수 있어야 한다. 개발에 어려움이 발생하면 같이 고민하고 해결해 줄 수 있는 좋은 동료가 돼야 한다.

02

개발자들과 소통하는 법

소통의 문화 이해하기

관리자가 아닌 파트너 되기

회사에서 어려움을 겪고 있는 직원과 높은 성과를 내는 직원이 있을 때 누구에게 관심을 가져야 할까? 일반적인 관리자는 어려움을 겪고 있는 직원에게 더욱 관심을 가질 것이다. 하지만 냉정하게 생각하면 높은 성과를 내는 직원에게 관심을 주는 게 회사 전체 성과를 향상하는데 도움이 된다. 우수한 직원에게 주는 관심과 격려는 더욱 좋은 성과를 끌어낼 수 있다.

우리가 뽑은 개발자들이 새로운 서비스 구축을 완료했거나 우수한 성과물을 내었다면 어떻게 해주면 좋을까? 이들은 회사를 위해 많은 노력을 했으므로 격려와 칭찬, 그리고 보상까지 해주는 것이 좋다.

만일 곤란을 겪는 직원들을 보살피느라 우수한 성과를 낸 직원을 소홀히 한다면 어떻게 될까? 높은 성과를 낸 직원으로서는 회사에 상

당한 기여를 했음에도 홀대받는다고 생각할 것이다. 혹은 역차별이라고 생각할 수도 있다.

반면 사소한 부분을 일일이 간섭하고 통제하여 모두를 피곤하게 한다면 어떤 결과를 가져올까? 너무 많은 간섭과 억압은 역효과를 불러올 수밖에 없다.

우리가 소중하게 뽑은 개발자들에게 지나친 간섭 또는 무관심을 보여주어서는 안 된다. 적절한 관심과 배려를 통해 동반자가 되는 것이 좋다. 특히, 우수한 성과를 낸 개발자에게는 적극적인 보상과 격려를 잊지 말아야 한다.

우수한 직원은 모두 좋은 평가를 받을 수 있게 하라

성과평가는 기업 입장에서 반드시 필요하다. 일반적인 기업에서는 성과평가 시 상대평가를 활용한다. 하지만 많은 직원은 이러한 성과평가 체계가 잘못되었다고 생각할 수 있다. 특정 부서나 집단이 유리하도록 성과평가가 설계될 수 있기 때문이다. 한 설문 조사에 따르면 미국 직원 중 71%는 성과평가의 공정성에 문제가 있다고 생각했다. 이렇듯 누구나 만족할 수 있는 성과평가 제도를 만들기란 정말 어려운 일이다.

그렇다면 개발자를 평가할 때는 어떤 방식을 활용하는 것이 좋을까? 개발자 평가를 객관적으로 하기 위해서는 성과를 정량화해야 하는데 이것이 어려운 경우가 많다. 과거에는 얼마나 회사에 오래 앉아 있는지, 야근은 얼마나 했는지 등으로 직원을 평가하던 시절도 있었

다. 하지만 시대가 바뀌었고 요즘 이렇게 했다가는 난리가 날 것이다.

가장 객관적으로 평가하기 위해서 평가 기준을 마련해야 하나 모두가 인정할 수 있는 기준을 만들기가 쉽지 않으므로 다음과 같은 방법을 추천한다.

팀 전체가 열심히 하여 성과를 올린 경우 팀, 혹은 직원 모두에게 높은 점수를 부여하는 것이 좋다. 줄 세우기 식 평가가 아닌 절대평가 방법을 활용하는 것이다. 마이크로소프트에서는 직원 개개인에 대한 평가보다는 협력을 통한 성과 창출에 중점을 둔다. 혼자 하는 업무보다 협력을 통한 업무가 더욱 큰 효과를 낼 수 있기 때문이다. 이러한 점수 평가가 보상으로 이어지면 더욱 효과적일 것이다. 팀 위주의 공평한 평가 제도를 통해 개발자들은 순위에 연연하지 않고 본인의 동기를 기반으로 각자의 자리에서 최선을 다할 것이다. 이것이 협력으로 연결되고 우리 회사의 성과로 이어지는 것은 당연하다.

개발자의 전문성을 인정하고 활용하라

개발자들이 가진 전문성이 많이 발휘될수록 좋은 서비스가 나올 확률이 높아진다. 이들이 가진 전문성을 충분히 이해하고 격려하는 것이 좋다. 특정 부서에서 개발 업무를 담당하는 시니어 급의 개발자가 있다면 이들의 전문성을 충분히 발휘할 기회를 제공해야 한다. 회사의 중요한 의사결정에 참여시킬 수도 있고, 교육 부서에서 다른 개발자들에게 지식을 전수할 수도 있다. 사실 전문가는 다른 사람에게 본인의 지식을 전달할 때 가장 빛난다. 따라서 직원 교육을 위해 엉뚱

한 직원을 뽑아 회사를 망치는 것보다는 내부에서 오랜 경력을 가진 개발자의 전문성을 활용하는 것이 바람직하다.

개발자들을 존중하라

개발자들이 가치 있다고 느끼는 방향으로 개발 및 업무를 수행할 수 있도록 해주자. 앞서 언급한 바와 같이 개발자가 본인이 하는 일에서 의미를 찾을 수 있도록 도와주어야 한다. 또한, 개발자들은 일반 직원들과는 다른 성향을 갖고 있다는 것을 이해해야 한다. 이들이 과거의 경험을 기반으로 의사결정을 할 때도 존중해주는 것이 필요하다.

경력직으로 입사한 개발자는 이전 회사에서 다양한 업무를 경험했기 때문에 효율적인 업무 방법을 잘 알 수 있다. 이들이 효율성 향상을 위해 제시하는 의견들을 적극적으로 수용하고 업무에 반영하는 것이 좋다. 이들은 회사에서 인정 및 존경받는 개발자가 된 듯한 느낌을 받을 수 있을 것이다. 이는 결국 각자 분야에서 오랫동안 근무할 수 있는 환경을 조성해주는 방법이다.

지속적인 도전 기회를 주어라

개발자들은 학습에 대한 욕구가 매우 강하다. 이들이 끊임없이 배울 수 있도록 도움을 주어야 한다. 이 학습은 업무와 직접적인 관련이 있을 수도 있고, 그렇지 않을 수도 있다. 이들이 기술 습득을 위해 도서 구입을 원하거나 강의, 세미나에 참가하고 싶다고 하면 언제든지 아낌없이 지원해야 한다. 또한, 이들의 새로운 도전을 적극적으로 장려해

야 한다. 새로운 서비스를 만들 때 이런저런 시도를 해보곤 하는데, 개발자들은 이러한 시행착오를 통해 경험을 쌓고, 이것이 실력으로 이어진다. 이러한 다양한 시도와 도전을 팀원들에게 공유하고 서로 장려하는 문화를 만들어야 한다.

개발자에게 다양한 방법으로 도전 기회를 주는 것은 매우 중요하다. 이는 개발자가 우리 회사에서 '일의 의미'를 찾을 수 있도록 도와주는 것이며 결과적으로 회사의 서비스 품질 향상과 연계되기 때문이다.

다양한 스타일로 팀과 소통하는 방법

우리는 개발자 개인 혹은 팀과 주기적인 소통을 해야 한다. 이들이 하는 말을 귀담아들어야 어떤 문제가 발생할지, 해결은 어떻게 해야 할지를 알 수 있다.

우선 업무에 관한 소통이 필요하다. 매주 할 수도, 격주로 할 수도 있다. 주기는 회사 사정에 맞춰서 진행하면 된다. 그렇다면 어떤 소통 방식을 활용하는 것이 좋을까? 여러 방식이 있을 수 있는데 그중 팀원들이 선호하는 방식으로 진행하는 것이 좋다. 업무에 관한 미팅이 필요하다면 공식적인 팀 회의를 통해 현재 프로젝트의 진행 상황, 지난 미팅 후의 피드백 결과 확인, 향후 개발 일정에 관한 이야기를 심층적으로 하는 것이 좋다. 무겁지 않은 형식으로 회의를 하면 훨씬 가벼운 마음으로 참석할 수 있을 것이다.

개발자가 느끼는 어려움에 대해 주기적으로 소통하는 시간도 필요

하다. 개발 진행에 어려움이 있다면 적극적으로 공감해줘야 한다. 프로젝트를 수행하면서 시간이 많이 소요되는 부분, 복잡한 부분, 잘 해결되지 않는 부분 등의 내용을 자유롭게 이야기할 기회를 제공하고, 회사는 이에 대해 깊게 이해하고 공감을 표현하는 것이 좋다. 이를 통해 개발자들은 회사에 대해서 더 깊은 신뢰를 하게 된다. 이러한 문화가 회사의 발전에도 긍정적인 영향을 줄 것이다.

인간 관계를 강화할 수 있는 소통의 시간도 필요하다. 회사의 동료이지만 서로의 인간적인 모습을 이해할 수 있는 시간이 될 것이다. 업무적인 대화보다는 개인적인 관심사 위주로 소통하는 것이 좋다. 자전거에 관심이 많다면 자전거 여행 코스, 자전거 구입법 등을 주제로 대화를 진행하면 인간적으로 더욱 친밀해질 수 있다. 이러한 친밀감은 회사에서 업무를 추진할 때 인간적으로 진행할 수 있게 만들어준다. 당연히 회사의 성과 향상에도 기여할 수 있을 것이다.

서버 컴퓨터 기업인 액세스랩은 어떤 방법으로 소통할까? 액세스랩에는 회식이 따로 없다. 대신에 상반기, 하반기 워크숍을 열어서 회사 문화를 함께 만들기 위해 노력한다고 한다. 또한, 회사 대표와 카카오톡, 슬랙 등의 채널을 통해 커뮤니케이션을 수행한다고 한다. 이를 통해 회사의 어느 직원도 대표에게 다양한 의견을 건의할 수 있는 것이다. 식사는 회사에서 비용을 지급하고 함께 먹는 문화를 권장한다고 한다. 식사하는 자리에서는 개인적인 이야기를 많이 하므로 서로 더 친해질 수 있는 계기가 되기 때문이다. 특정 개발자 1~2명이 특별하다는 느낌을 주지 않으려고 노력한다고도 한다. 비 개발자 직원도 개발

회의에 함께 참여하게 하고 관련 내용을 알기 쉽게 설명하여 전체 업무 흐름에 대해서 이해도를 높인다고 한다. 이런 시도는 개발 업무가 특징인 이 하는 것이 아닌, 회사 전체에서 함께 하는 업무라고 생각하게 되어 서로 간의 팀워크를 높이는 장점이 있다.

상호 간 피드백 문화 정착하기

지속적인 피드백이란 무엇일까? 개발자가 하는 일에 계속 간섭하는 것인가? 아니면 지금 하는 일이 긍정적으로 흘러갈 수 있도록 지속해서 도움을 주는 것인가? 사실 피드백, 성과평가 등은 개발자들이 좋아하지 않는 단어이다. 그렇지만 조직의 발전을 위해서 피드백은 반드시 필요하다. 그러므로 상호 간에 피드백하는 방법을 아래와 같이 다양하게 시도해보는 것이 좋다.

피드백은 긍정적인 피드백과 잘못된 점을 바로잡는 피드백(이하 개선점 피드백)이 있다. 당신이라면 어떤 피드백을 듣고 싶은가? 내가 밤을 새워서 프로젝트의 일부를 완성도 높게 끝낸 경우, 해당 업무에 대해 긍정적인 피드백을 받게 되면 기분이 어떨까? 긍정적인 피드백은 받는 사람과 주는 사람 모두가 기분 좋아진다. 피드백을 주는 사람 입장에서는 개선점 피드백보다 말을 꺼내기 훨씬 쉬우니 그 효과는 몇 배가 된다. 다만, 잘하지 않았는데도 억지로 긍정적인 피드백을 줄 필요는 없다.

개선점에 대한 피드백도 반드시 필요하다. 사람은 일하다 보면 분명 실수를 할 수 있다. 본인만의 스타일대로 개발과 업무를 하다 보면 빠

진 부분도 있을 수 있다. 이를 제때 바로잡아 주지 않으면 나중에 이것을 고치는데 시간이 더 걸리게 된다. 반면 적시에 피드백을 해주면 빠른 개선을 통해 우수한 결과물이 나올 확률이 높아진다.

이러한 지속적인 피드백을 위해 가장 중요한 것은 무엇일까? 개발자 개개인에 대해 소통하고 이해하려는 노력이 필요하다. 즉, 꾸준한 관심을 가지고 이들이 어떤 상황에서 업무를 하는지, 어려움은 없는지를 주기적으로 파악해야 한다. 그러면 개개인에 대한 피드백을 더욱 객관적으로 수행할 수 있으며 성과 향상에도 큰 도움을 줄 것이다.

수평적인 문화 사례

개발자들은 서로 협력하며 일을 하기도 하지만 독립적인 공간에서 일하는 것을 더욱 선호한다. 이러한 문화는 개발자들이 수평적인 문화를 좋아한다는 것을 대변한다. 최근 많은 기업들은 부장, 팀장, 과장 등으로 이어지는 수직적인 문화에서 탈피해 직급 없이 모두가 같은 직급을 갖는 수평적인 문화로 바꾸고 있다.

교육 플랫폼을 운영하는 클래스101이라는 스타트업은 사장부터 직원까지 모두 반말(뽕어)을 사용한다고 한다. 서로 간의 호칭은 별명을 부르며 직급은 대표와 부대표만 있고, 다른 직원들은 모두 리드라는 직책으로 불린다. 이러한 독특한 기업 문화는 더욱 창의적인 아이디어와 자유로운 의견 개진이 가능하여 기업 내 수평적인 문화 형성에 도움을 준다.

모바일 헬스케어 스타트업인 힐링페이퍼는 서로를 부를 때 각자 정

한 호칭(닉네임)을 쓴다. 각 팀의 리더가 있지만 업무가 수직적이지 않고 수평적으로 진행된다. 시니어의 의견만 중시되는 것이 아니라 주니어 개발자들의 아이디어도 적극 반영한다. 주니어 개발자가 좋은 아이디어를 제시하면 시니어 개발자는 격려와 더불어 개인 프로젝트로 진행할 수 있도록 지원해준다. 또한 채용 과정에서 상급자들의 의견보다는 같이 일할 팀원들의 의견을 더욱 중요시한다. 각 팀에서 이력서를 검토 후 의견을 내면 이들의 결정을 가장 중시하여 채용을 진행한다고 한다.

카셰어링 기업인 쏘카는 이름 대신 닉네임을 사용하고 있으며 수평적인 문화 정착에 앞서고 있다. 신입 개발자가 적극적인 의견을 내는 것도 장려하는 분위기이다. 이들은 번뜩이는 아이디어를 쏟아내 회사의 문화를 수평적으로 바꾸는 데 일조하고 있다.

예시로 든 기업 말고도 많은 기업들이 수평적인 문화를 만들고자 노력하고 있다. 우리 회사가 수직적인 문화를 갖고 있다고 판단한다면 지금 당장 개선을 위한 노력을 시작해야 한다.

신뢰 관계 만들기

신뢰 관계 형성의 중요성

어제 입사한 개발자에게 갑자기 “여자 친구는 있어요?”, “이번 주말에 뭐해요?” 등의 개인적인 질문을 하면 그는 어떤 생각을 할까? “별로 친하지도 않은데 왜 저런 질문을 하지?”라고 생각할 수 있다. 개인

적인 질문은 서로의 신뢰 관계가 어느 정도 형성된 이후에 하는 것이 자연스럽다.

사람 간의 신뢰 관계는 하루아침에 만들어지지 않는다. 신뢰 관계는 시간에 어느 정도 비례한다고 할 수 있다. 이는 꾸준한 관계 유지와 상호 존중, 각자의 행동 패턴 등 다양한 요소들을 기반으로 형성된다. 이러한 관계는 처음에 형성하기는 어렵지만 한번 형성되면 쉽게 무너지지 않는다. 회사 내에서 개발자와 신뢰 관계가 형성되면 이 고리는 아주 단단해진다. 개발자들은 회사와 형성된 신뢰 관계를 바탕으로 '일의 의미'도 쉽게 찾을 수 있을 것이다. 개발자들이 높은 신뢰 관계를 기반으로 업무에 임한다면 회사의 생산성은 자연스레 높아질 것이다.

캐슈어링 기업 쏘카는 작은 성공사례를 기반으로 신뢰 관계를 형성하였다. 쏘카에서 처음 개발팀을 운영할 때에는 팀이 1개밖에 없어서 협업할 일이 없었다. 지금은 5개 규모의 부서를 운영하며 서로의 팀과 협업을 진행하고 있다고 한다. 개발자들은 작은 프로젝트부터 함께하며 점점 큰 프로젝트로 규모를 키워가며 상호 간 신뢰를 형성했다. 작은 프로젝트부터 성공을 경험하며 회사 및 동료들과 자연스레 신뢰 관계를 형성하게 된 것이다. 이렇듯 개발자들이 성공할 수 있는 다양한 기회를 제공하는 것도 회사의 역할이다.

이러한 신뢰 관계를 만들어나가는 데 중요한 역할을 하는 것은 무엇일까? 바로 팀워크다. 큰 규모의 기업은 팀워크를 객관적으로 평가할 수 있는 시스템이 있지만 작은 규모 기업은 이러한 시스템을 갖추기가 쉽지 않다. 규모가 작은 회사에서는 우수한 역량을 가진 개발자가

혼자 개발하는 것보다 전체 팀이 평균 이상의 성과를 내는 것이 더 중요하다. 실제로 이렇게 서로 이해하고 함께하는 팀워크를 구성했을 때 회사의 성과는 좋아지게 된다.

이러한 팀워크가 좋아지게 하려면 어떻게 해야 할까? MZ세대라고 불리는 요즘의 젊은이들을 상대로 예전처럼 회식을 많이 한다고 팀워크가 좋아지지 않는다. 또한, 회사가 성장해야 개인이 성장한다는 말도 통하지 않는다. 예전에는 내가 협력을 잘해야 회사가 발전하고 개인이 성장할 수 있다는 인식이 강했다. 지금 세대는 반대로 내가 성장해야 회사가 발전한다고 생각한다. 이러한 세대의 특성을 먼저 이해하고 이를 기반으로 팀원들과 업무를 해야 한다. 이들과 팀워크가 좋아지는 중요한 미덕은 소통이다. 쉽게 말하면 회사 대표는 항상 신입 사원의 새로운 제안을 받아들일 준비가 되어있어야 한다. 다양한 방법으로 소통하면서 이들과의 신뢰 관계를 구축하면 결과적으로 좋은 팀워크와 회사의 성과로 이어지게 된다.

개발자들이 자유롭게 일하도록 하라

신뢰 관계 구축에 전혀 도움이 되지 않는 행동은 회사 내에서 권한을 행사하려는 것이다. 회사에는 대표나 이사, 팀장 등 주요 결정 권한을 가진 사람들이 있다. 신뢰 관계 형성을 위해 이들이 가진 권한을 어느 정도 내려놓을 필요가 있다. 누구나 이런 권한을 누리고 싶어 할 것이다. 하지만 이러한 권한에 초점을 맞추면 조직이 건강해질 수 없다. 권한보다는 관계에 초점을 맞추어야 한다.

구글은 대표적인 개발자 중심의 회사이다. 구글에서는 관리자들이 가진 일방적 권한을 내려놓도록 했다. 그리고 직원의 관리보다는 관계에 초점을 맞추었다. 구글이 내놓은 '좋은 관리자가 가져야 하는 8가지 조건'을 보면 어떤 관리자가 개발자에게 좋은 관리자인지 잘 알 수 있다.

좋은 관리자가 가져야 하는 8가지 조건

- ① 좋은 코치다.
- ② 팀에게 권한을 양도하며 마이크로 매니지를 하지 않는다.
- ③ 팀원의 성공에 관심을 표명하며 개인적 삶에도 관심을 기울인다.
- ④ 생산적이며 결과를 중심으로 사고한다.
- ⑤ 훌륭한 커뮤니케이션 능력을 가지고 있다.
- ⑥ 팀원들이 경력을 키워나가도록 도움을 준다.
- ⑦ 팀을 위한 명확한 비전을 가지고 있다.
- ⑧ 팀에게 조언을 해주기에 충분한 기술적인 능력을 갖추고 있다.

이를 간략하게 설명하면 개발자들을 사소하게 통제하려고 하지 말고, 이들이 스스로 결정 권한을 갖고 자유롭게 일을 할 수 있도록 믿고 맡기라는 것이다. 쉽게 말하면 관리자의 권한은 최소화하고 개발자들의 자유를 극대화하는 것이다. 개발자들이 편하게 일할 수 있는 환경을 만들어주고 이들이 회사에서 자유롭게 일하는 느낌을 받게 해야 한다. 개발자에게 자유를 준다는 것은 마음대로 하라는 얘기가 아니다.

자유가 있으면 책임도 있기 마련이다. 일을 하는 방법, 소통 방법, 근무 방법 등에서 회사에 얽매어 있다는 생각이 들지 않도록 해주는 것이 좋다.

개발자와 올바른 방법으로 소통하라

개발자들과의 신뢰 관계 구축을 위하여 가장 중요한 건 주기적인 소통이다. 업무를 수행하다 보면 여러 상황에 부딪히기 마련이고 회사에 요구해야 할 것도 많아진다. 이러한 이야기를 들을 수 있는 시간을 따로 만드는 일은 쉽지가 않다. 하지만 일부러라도 주기적인 대화 시간을 만들어야 한다. 일대일 형식도 좋고 일대다 형식도 좋다. 각 기업의 문화가 다를 것이므로 어떤 형식이 좋다고 딱 잘라 말하기는 어렵다. 중요한 것은 서로 만나서 대화하는 시간을 가지라는 것이다. 신뢰 관계 형성에 대화와 소통만큼 중요한 것은 없다. 하지만 개발자의 특성을 알지 못하고 소통을 하면 오히려 독이 될 수 있다.

그렇다면 개발자들과는 어떻게 소통하는 것이 좋을까? 여러분이 가장 궁금해할 질문일 것이다. 한 대표가 다음과 같이 말했다. “OO 개발자님, 우리 회사에서 고객 추천 시스템을 개발하려고 합니다. 고객이 만족할만한 멋진 서비스로 화려하게 만들어보세요”.

개발자는 이 말을 듣고 무슨 생각을 할까? “대체 뭘 어떻게 하라는 거야?”라고 생각할 확률이 100%이다. 개발자들은 추상적인 표현을 좋아하지 않는다. 추상적으로 말한다는 것은 기술을 제대로 모른다는 뜻이다. 이런 식으로 얘기하는 사람을 신뢰하지 않게 된다. 개발자들

은 구체적이고 명확한 단어를 사용하여 대화하는 것을 좋아한다. 개발자들과 대화하는 방법을 한마디로 요약하면 명확한 ‘입력’과 ‘출력’이다. 즉, 입력에 해당하는 ‘고객의 요구사항’, ‘담당할 개발 범위’, ‘개발 기간’, ‘대상 고객’ 등을 구체화하고 출력에 해당하는 ‘디자인’, ‘사용자 인터페이스’ 등을 명확하게 알려줘야 한다.

03

팀의 성장은 회사의 성장이다

개발팀 운영하기

시스템 표준을 정하라

개발팀은 혼자 잘해서 되는 것이 아니고 협업 능력이 중요하다. 서로 간의 소통을 위해서 최적화된 시스템이 있어야 한다. 개발자들은 협업을 위해 이용하는 시스템이 자주 바뀌거나 업무 자체가 시스템화 되어 있지 않은 것을 싫어한다. 그러므로 개발 프로세스와 시스템 표준 마련을 위한 꾸준한 노력이 필요하다.

먼저 개발 프로세스를 명확하게 시스템화할 필요가 있다. 업무를 추진하는 방식이나 피드백 방식도 명확히 해야 좋다. 이를 위해서 다양한 툴을 활용한다.

소통을 위해서는 Notion과 같은 협업 툴을, 문서 작업은 Google docs를, 코드 리뷰는 깃허브를 활용하는 것이다. 이러한 툴을 활용하여 피드백은 어떻게 진행하는 것이 좋은지, 협업 방식은 무엇인지를 구

체화해야 한다.

팀의 표준 업무 방식을 만드는 일도 중요하다. 이러한 방식을 통해 코드 리뷰나 설계 단계에서 활발한 의사소통이 가능하다. 코드를 변경할 때마다 테스트는 얼마나 할지, 코드 리뷰와 디버깅은 어느 시점에서 할지 등을 명확하게 정해야 한다. 구체적인 소통의 시점도 명확하게 정해야 한다. 코드 리뷰는 매주 언제 하며, 팀 회의는 언제 하는지를 명확하게 정해서 팀원 모두가 지킬 수 있도록 하자. 다시 한번 강조하지만 불분명한 것들을 모두 명확하게 바꾸는 작업이 필요하다.

코드 리뷰하기

코드 리뷰는 코드를 작성한 개발자 본인이 아닌 다른 사람이 점검해주는 일이다. 실력이 아무리 우수하더라도 본인의 코드 오류나 실수를 잡아내기는 어렵다. 코드 리뷰를 통해 팀원들은 담당자가 놓친 부분이나 오류 사항을 점검한다. 코드의 효율성 향상 방법과 더욱 경제적인 코드를 짤 수 있는 방법 등의 다양한 의견을 줄 수도 있다.

코드 리뷰를 하면 어떤 장점이 있을까?

첫째, 개발자가 흔히 하는 실수를 빠르게 고쳐줄 수 있다. 아주 사소한 실수도 코드가 길어지면 잡아내기 어렵다. 이로 인하여 개발 결과물의 품질이 저하될 수 있다. 코드 리뷰를 통해 조기에 발견하여 수정하는 것이 중요하다.

둘째, 여러 개발자가 정해진 표준에 따라서 개발을 진행할 수 있어 결과물의 일관성이 높아진다. 서로 간의 리뷰를 통해 각자가 진행하는

코드의 스타일을 알 수 있고 지속적으로 확인함으로써 추후 코드 통합 등에 매우 유리하다.

셋째, 중복되거나 불필요한 코드를 미리 검출할 수 있다. 코드가 길어지면 중복되거나 불필요한 부분이 분명 발생하는데, 정작 본인은 이를 발견하지 못하는 경우가 많다. 다른 개발자가 이를 보면 쉽게 찾아낼 수 있으므로 리뷰 과정은 반드시 필요하다. 이 과정을 통해 재사용이 가능한 모듈을 중복으로 개발하는 상황도 막을 수 있다.

넷째, 긴급한 상황에서 담당자 부재 시 빠른 대응이 가능하다. 팀원들이 서로 간의 코드를 이해하고 있으므로 개인 사유로 담당자가 없을 때도 신속한 대응이 가능하다.

코드 리뷰는 어떻게 하는 것이 좋을까? 리뷰 방식은 온라인 리뷰를 활용하면 좋다. 굳이 만나지 않아도 충분히 의견 교환이 가능하다. 정말 중요한 코드를 리뷰할 때에는 직접 모여서 의견을 공유하는 방식을 추천한다. 코드 리뷰는 주기적으로 하는 것이 좋다. 일주일에 한 번 정도는 서로 작성한 코드를 리뷰하도록 하자. 다른 사람에게 보여주기 부끄러운, 혹은 숨기고 싶은 코드도 서로 리뷰할 수 있도록 장려하는 것이 좋다.

다만, 코드 리뷰 시에 주의할 점이 있다. 코드 리뷰의 목적은 제품과 서비스를 개선하고 회사와 우리 모두의 성장을 위한 것임을 명확하게 알릴 필요가 있다. 개발자는 개선 의견이 왔을 때 감정적으로 대응하면 안 된다. 리뷰어도 순수하게 코드만 보고 리뷰를 해야 한다. 평소의 감정을 담은 리뷰를 하면 안 된다. 그러므로 일종의 원칙을 세우는 것

이 좋다. 예를 들면, 코드 리뷰는 공개적인 장소(GitHub)에서만 진행하고 개인적인 의견을 직접 전달하는 방식은 금지할 수 있다. 또한, 리뷰 시에 사적인 감정을 최대한 배제하고 객관적인 리뷰를 하는 등의 원칙을 세우는 것이 좋다. 그렇지 않으면 서로 간의 감정 소모만 심해질 뿐이며 긍정적인 팀 분위기 형성에도 도움이 되지 않는다.

팀의 현재 상황을 지속적으로 살펴라

개발팀이 잘 운영되고 있는지 지속적으로 관심을 가지고 살펴야 한다. 어떤 문제가 발생하고 있는지, 코드는 주기적으로 잘 나오는지, 팀원들 사이에 마찰은 없는지 등을 주기적으로 확인할 필요가 있다. 이를 위한 몇 가지 방법을 소개한다.

우선 해당 팀의 코드 릴리스(배포) 빈도에 관심을 가져야 한다. 코드 릴리스를 자주 하지 않는 팀을 살펴보면 문제가 발생하고 있을 확률이 높다. 물론 자주 릴리스할 수 없는 서비스도 있지만 대부분의 서비스는 이런 공식 적용이 가능하다. 개발팀의 팀장은 어떻게 하면 팀을 효율적으로 운영할 수 있는지, 업무 배분은 어떤 방식으로 해야 하는지를 잘 알아야 한다. 이를 바탕으로 신속한 코드 릴리스를 위해 끊임없이 노력해야 한다. 사실 코드 릴리스를 하는 과정은 시간이 오래 걸릴 수밖에 없다. 그러한 상황에도 팀 내 적절한 역할 배분과 빠른 업무 결정 등을 바탕으로 효율적인 릴리스가 가능해야 진정한 팀장으로서 역할을 했다고 판단할 수 있다.

다음으로 문제 발생 빈도를 줄이는 방안에 대해 고민해야 한다. 우

리의 목표는 서비스의 품질을 높이고 안정적으로 운영하는 데 있다. 발생할 문제를 최소화하는 것도 품질 관리에 속한다. 개발자는 안정적인 상황에서 근무하기를 원한다. 지속해서 문제가 발생하면 이를 해결하는데 수많은 시간을 쏟아야 한다. 이는 전체적인 일의 효율을 크게 낮춘다. 물론 문제가 발생하지 않을 수는 없다. 그래도 최소화하기 위해서 여러 노력을 해야 한다. 또한, 개발자들이 단순 문제 해결에 너무 많은 시간을 쏟지 않도록 한다. 해당 문제 해결로 인하여 개발자 본인과 팀이 발전한다면 더할 나위 없이 좋지만 단순히 문제만 찾아서 해결하고 끝나버리는 식의 일은 우리 조직의 발전에 아무런 도움이 되지 않는다.

마지막으로 팀원들 간의 관계와 업무 상황을 주기적으로 파악하는 것이 좋다. 개발자들 모두 각자 개성이 강하다. 그러므로 이들 간에 마찰이 발생하는 것을 심심치 않게 보았다. 마찰이 발생하지 않도록 예방하는 것이 중요하다. 혹시나 이러한 상황이 오더라도 현명하게 대처해야 한다. 마찰은 업무 배분, 코드 리뷰, 지속적인 오류 발생 등 다양한 상황에서 발생한다. 팀의 리더는 갈등 상황에서 구성원들의 각자 성향을 명확히 파악한 후 이에 맞는 해법을 찾아줘야 한다. 그리고 불필요한 업무로 인해 업무과 과중한 건 아닌지도 주기적으로 파악해야 한다. 불필요한 업무로 인해 본연의 업무에 치중하지 못하는 상황을 막아야 하기 때문이다.

문제가 있는 팀 개선하기

앞서 언급한 것과 같이 코드 릴리스 빈도가 낮거나 팀원 간 관계가 엉망인 팀을 운영해야 할 때도 있다. 같이 일하는 팀원들의 원성이 자자하고 팀 리더도 불만이 많을 수 있다. 이 팀은 분명히 잘못 운영되고 있지만, 누구도 무엇이 잘못되었는지 확실한 판단을 내릴 수 없는 상황이다.

가장 큰 문제는 팀원들 간의 관계가 좋지 않은 것이다. 사람들이 여러 유형이 있듯이 개발자들도 여러 유형이 있다. 이중 매사에 부정적으로 임하는 사람은 어떻게 대하면 좋을까? 그가 잘못하고 있는 부분을 명확하게 지적하고 개선할 수 있도록 요구해야 한다. 부정적인 분위기를 하나만으로 팀원들은 매우 피곤해지며 하루의 에너지를 다 빼앗긴다. 그러므로 신속히 개선을 요구해야 한다. 그래도 개선되지 않는다면 다른 팀으로 이동시키는 게 그나마 나은 방법이다. 사실 사람은 잘 바뀌지 않는다. 우리 팀과 맞지 않는다면 다른 길로 갈 수 있도록 도와주는 것이 서로 좋은 방법이다.

팀원들이 과도한 야근과 업무로 스트레스를 받는 경우는 어떻게 해결해야 할까? 개발 업무 이외의 다른 잡무들로 고통받고 있을 수 있다. 이런 경우는 해결이 간단하다. 잡무를 담당할 다른 직원을 채용하면 된다. 업무가 비교적 적은 팀원들에게 분배하는 방법도 있다. 한 사람이 과중한 업무에 시달리지 않도록 관리를 해야 한다.

사실 긴급한 서비스 개선 작업 등이 필요할 때는 어쩔 수 없는 경우가 많다. 이럴 때는 옆에서 도움을 주는 동반자의 역할을 해야 한

다. 우리 팀원 전체가 같이 고생하고 있다는 느낌을 받게 해주어야 한다. 바쁜 상황에서 팀 리더가 휴가를 가거나 조기 퇴근을 해버리면 팀원들은 업무에 집중할 수 없을 것이다. 같이 고민해주고, 도와주며, 이 상황이 끝나면 회사 차원의 보상이나 휴가 등이 있을 것이라고 독려하는 것도 하나의 방법이다. 맛있는 저녁 식사와 간식도 풍부하게 제공하자. 이들에게 진심으로 감사하는 마음을 갖고 응원해줄 수 있어야 한다. 이번 상황만 모면하면 된다는 안일한 생각은 하지 말아라.

서로 간의 협업이 잘 안 되는 경우도 있다. 분명 잘하고 있는 것처럼 보였는데 팀원들의 손발이 맞지 않는다. 그로 인해 코드 릴리스가 늦어지고 이에 대한 불만이 쌓이는 경우는 어떻게 해야 할까? 이런 상황은 사실 가장 해결하기 어렵다. 때문에 우선 장기적인 관점에서 바라볼 필요가 있다. 이 상황에서 가장 필요한 건 바로 소통이다. 팀원 간에 소통할 기회를 자주 만들어 서로의 문제점을 찾아 나가는 것이다. 사적으로 친해질 기회도 만들면 좋다. 회사 업무를 떠나 인간 대 인간으로 만나서 함께 식사하고 영화도 보는 기회를 만들어보자. 단, 업무 시간을 벗어나서 이런 것을 하면 요즘 개발자들은 정말 싫어한다. 한 달에 하루 정도는 회사에 나오지 않고 영화관에서 만나는 방법을 써보자. 그날은 원래 근무일이어야 한다. 출근 대신에 소풍을 가는 마음으로 올 수 있도록 말이다. 이런 자연스러운 방법으로 직원들 간의 관계를 개선하면 협업 관계 향상에 많은 도움을 줄 수 있다.

360도 평가하기

모든 회사의 숙명인 평가를 해야 하는 시점이 다가온다. 모든 개발자가 잘했으므로 같은 평가를 주고 싶지만 현실은 그렇지 못하다. 여기에서 360도 평가 방법을 소개한다. 이 방법은 360도 평가, 360도 다면평가 등의 여러 이름으로 불린다. 360도 평가는 상사가 부하를 일방적으로 평가하는 방식이 아닌 다양한 사람들의 의견을 바탕으로 정보를 수집하고 평가하는 방식이다. 여기서 다양한 사람들이란 자기 자신 뿐만 아니라 동료, 상사, 부하 등 우리가 회사에서 자주 접하는 사람들을 말한다.

이 평가 방법은 여러 사람의 피드백을 바탕으로 이루어지므로 시간이 꽤 많이 소요된다. 그렇지만 예상보다 많은 정보를 얻을 수 있다는 장점이 있다. 이러한 정보는 꽤 정확한 편이다. 한두 명의 평가가 아닌 주변에서 접하는 대부분의 사람을 활용하기 때문이다.

360도 평가 방법의 성공적인 운영을 위해서 몇 가지 기억해야 할 점들이 있다.

첫째, 평가 항목의 커스터마이징(Customizing)이 필요하다. 다른 회사의 평가 항목을 가져다 쓰는 것이 아니라 우리 회사만의 평가 항목을 만들어야 한다. 특히, 부서별 특색이 다른 경우 이러한 것을 적극 반영할 필요가 있다.

둘째, 최근 1년간의 성과를 바탕으로 평가한다는 것을 잊지 않도록 한다. 사람들은 일반적으로 3개월 이상 지나면 그때의 일은 잘 기억하지 못한다. 만약 연초에 엉망으로 개발을 했던 사람이 후반부에 아주

우수한 개발 실적을 냈다고 해보자. 사람들은 앞에 있었던 일보다는 최근 일을 더 잘 기억하기 때문에 우수한 평가를 줄 수도 있다. 이러한 문제를 피하기 위해서는 협업 시 기록을 철저히 해놓는 것이 중요하다. 그리고 과거의 이메일 등을 확인하여 월별로 어떤 협업과 소통을 했는지 파악해야 한다.

셋째, 모든 이해 관계자가 참여하는 것이 좋다. 말 그대로 다면평가이므로 해당 직원과 관계있는 모든 사람을 참여시켜야 한다. 내가 상대방을 성실하게 평가해야 상대방도 나에게 대해 성실하게 평가에 임한다는 생각으로 참여하는 것이 좋다.

넷째, 평가를 위한 준비부터 실행까지 충분한 시간을 할애해야 한다. 앞서 언급한 것처럼 이 평가 방법은 효율적이나 시간이 많이 소요된다. 여러 사람에게 평가서를 받아 취합하는 과정부터가 시간이 걸린다. 팀 리더는 취합한 평가서를 잘 읽어보고 요약 및 피드백을 작성해야 한다. 이러한 과정이 단 며칠 만에 이루어질 수 없으므로 여유를 갖고 준비하고 시작해야 한다.

다섯째, '평가 기준'을 공개해야 한다. 평가의 절차와 방법, 상위 평가의 기준 등을 정하고 구성원 전체에게 미리 공지해야 한다. 일반적으로 평가가 연말에 이루어지는 것을 고려해 연초에 해당 내용을 공지하면 좋다. 그래야 1년 동안 해당 평가 지표에 맞춰서 업무를 진행할 수 있고, 이에 따라 더욱 객관적인 평가가 가능하기 때문이다.

최근 기업들은 상대평가의 개념보다는 절대평가에 가까운 방법으로 직원 평가를 하고 있다. 즉, 직원 개개인에 대한 성과평가의 비중이

약해지고 있다는 뜻이다. 성과평가보다는 협력을 통한 성과 창출에 중점을 두는 시대로 변화하고 있다. 직원을 줄 세우려는 목적으로 평가를 활용해서는 안 된다. 직원을 평가하되 순위를 매기지 말고 추후 이들의 발전을 위해서 활용해야 한다.

팀 내 개발자들 대다수가 훌륭한 협업 방식과 퍼포먼스를 보여준다면 이들 모두에게 최고의 평가를 줄 수 있어야 한다. 또한, 성과급 등의 보상으로 이어진다면 동기부여에 큰 도움이 될 수 있다.

개발자가 성장할 수 있는 문화를 만들자

‘개인의 성장은 곧 회사의 성장이다’라고 아무리 강조해도 개발자들 본인에게는 와닿지 않을 수 있다. 말뿐인 경우가 많기 때문이다. 회사에서 하는 업무를 바탕으로 진짜 본인이 성장할 기회를 만들어줘야 한다. 회사가 직원을 소모품처럼 다루는 시대는 끝났다고 보면 된다. 특히 개발자들을 이런 식으로 대한다면 한 달을 못 넘기고 퇴사하는 사태가 발생할 수 있다.

개발자들이 성장할 수 있는 문화를 만들기 위해 회사는 다양한 노력을 해야 한다. 예를 들어 개인의 성장 방향을 명확한 경력 개발 로드맵을 통해 보여주고, 개인의 성과가 회사의 성과와 어떻게 연관되어 있는지 공유하는 것이 좋다. 이를 통해 개인은 자신이 하는 일이 의미가 있으며 자신의 지속적인 성장 가능성을 회사에 있다고 생각할 수 있다. 가끔은 자신이 하는 일이 회사의 결과와 매우 동떨어져 있다고 느낄 수도 있기 때문에 이 과정은 매우 중요하다. 또한, 고객의 목소리를

직접 들려주는 것도 직원에게 보람을 느끼고 공감할 수 있게 해주는 좋은 방법이다. 예를 들면, 회사 내 행사에 고객을 초청해서 우리가 개발한 것이 어떻게 도움이 되었는지를 듣고 다양한 사람들이 공감하는 모습을 보여주는 것도 좋은 방법이다.

떠나보내되 좋게 떠나보내라

개발자가 또 다른 성장을 위해 떠나려고 할 때도 세심하게 배려해야 한다. 우리 회사에서 가장 유능한 풀스택 개발자가 다른 회사로 이직하겠다는 결정을 통보해왔다. 회사 입장에서는 이를 어떻게 받아들여야 할까? 가기로 마음먹은 사람을 되돌릴 수는 없다. 이미 결정하기까지 많은 고민을 했을 테니까. 다만 이때가 중요하다. 누구보다 쿨하게 잘 보내주어야 한다. 다음 기회에 좋은 곳에서 다시 만나기를 기원하면서, 우리 가족이 좋은 곳으로 간다는 마음으로 보내주어라. 물론 회사 입장에서는 손실이 클 수 있다. 하지만 이미 벌어진 상황을 어찌겠는가? 우리가 잘 보내주어야 새로 들어올 개발자도 잘 들어올 수 있다. 생각보다 이 바닥은 좁다. 몇 번 건너면 다 아는 사이이다. 이 직원이 나갈 때 온갖 괴롭힘을 통해 고통스럽게 만든 후 퇴사시키면 어떨까? 내부 직원들은 이런 과정을 다 보고 있다. 이렇게 퇴사한 당사자는 뒤도 안 돌아보고 나갈 것이다. 나쁜 소문은 결국 화살이 되어 우리 회사로 날아올 것이다. 그럼 그만큼 유능한 개발자를 뽑을 수 있을 것 같은가? 이 바닥에서 한 번 그렇게 소문난 회사는 개발자들이 기피하는 회사 1순위가 되어있을 것이다.

아름답게 떠나보내 주어라. 그는 우리 회사의 좋은 기억을 잊지 못해 다른 회사에서 더욱 실력을 쌓은 후에 다시 돌아올 수도 있다.

퇴사자가 우리 회사를 좋게 떠나면 선순환이 된다. ‘이 회사에 가면 저 개발자처럼 성공해서 나올 수 있구나’, ‘저 회사에는 성장에 도움을 주는 시니어들이 많이 있구나’ 등의 좋은 소문이 외부 개발자들 사이에서 돌게 된다. 실제로 3~4년 차 경력을 가진 주니어 급 개발자 채용은 매우 어렵다. 이들은 팀에서 보고 배울 사람이 있는지를 확인하고 이직을 결정한다. 우리 회사에 좋은 팀이 유지되고 배울 수 있는 훌륭한 사람들이 많이 있다는 것을 알게 되면, 유능한 개발자들이 오고 싶어 하는 회사가 될 것이다. 즉, 잘 보낸 개발자 한 명이 열 명의 유능한 개발자를 불러오는 선순환 구조를 만든 것이다.

핵심 개발자가 떠난 후 우리 회사는 그냥 멍하니 있어야 할까? 절대 아니다. 이들이 왜 떠나게 되었는지 명확한 원인을 분석하고 이를 개선하기 위한 노력을 해야 한다. 그렇지 않으면 퇴사자와 같이 일했던 다른 개발자들도 하나둘 짐을 싸기 시작할 수 있다.

함께 일하는 방법

생각할 시간이 필요하다

아침에 출근해 팀 회의를 하고, 회의가 끝나면 점심을 먹고, 점심 후에 커피를 마시고 회사에 들어오면 피곤하다. 피곤하니 조금 쉬었다가

이제 개발을 시작하려고 하니 긴급회의 소집이 있다. 이 회의가 끝나니 집에 갈 시간이 다 되어간다.

우리가 쉽게 경험할 수 있는 개발자의 모습이다. 개발자에게는 오롯이 개발에만 쏟을 수 있는 시간이 필요하다. 하지만 위와 같은 일정으로는 개발할 시간이 없다. 앉아서 시작하려고 하면 회의해야 한다고 호출한다. 또 와서 시작하려고 하면 옆의 동료가 잠깐 부른다.

‘티타임도 업무’라는 말이 있다. 그만큼 회사에서 불필요한 시간을 보낼 때가 많다. 개발자뿐만 아니라 우리 대부분은 바쁘게 살며 생각할 시간을 갖지 못하는 경우가 많다. 하지만 생각할 시간이 없다는 것은 정말 불행한 일이다. 특히 개발자는 개발 업무를 효율적으로 수행하기 위해 고도의 집중력을 바탕으로 많은 생각을 해야 한다. 많은 고민을 통해야만 좀 더 좋은 코드와 결과물이 나오게 된다. 하지만 위와 같이 생각할 시간이 없다면 어떨까? 시간에 쫓겨 개발하게 되고 결과적으로 품질에 좋지 않은 영향을 미친다.

그러므로 개발자들이 불필요한 일에 시간을 쏟지 않도록 도와줘야 한다. 개발에 많은 생각과 고민을 할 수 있게끔 방해하지 않아야 한다. 생각을 많이 하라고 강요하는 게 아니라 그런 환경을 만들어주라는 얘기다. 요즘엔 재택 근무를 도입하는 기업이 많아지고 있는데, 이 또한 생각할 시간을 가질 수 있도록 도와주는 것이다. 어떤 기업은 개발자가 커피숍에서 일하고 싶으면 언제든지 커피값을 지원해주고, 회사 안이 아닌 커피숍에서 일할 수 있도록 배려해준다. 개발자 본인은 어디서 일해야 생산성이 가장 높은지 알고 있다. 개발자가 충분한 시간을

갖고 생각할 수 있는 시간을 가질 수 있도록 많은 배려가 필요하다.

회의 방법도 여러 가지이다

회사에 출근하니 바로 메신저가 울린다. “오늘은 신규 서비스 런칭과 관련된 회의를 진행합니다. 회의실로 모여주세요”. 아침부터 회의실에 팀원 모두가 모이게 되었고 바로 회의를 시작한다. 그런데 회의의 끝이 보이지 않는다. 회사에 출근하자마자 바로 회의를 시작했는데 점심 먹을 때까지 끝이 안 난다.

이런 상황에서 두 가지로 생각을 하는 그룹이 있을 수 있다.

첫 번째 그룹은 “이렇게 길게 회의를 해서 다양한 의견 수렴을 하니 좋다”고 생각할 수 있다(물론 긴 회의를 좋아하는 사람은 없다).

두 번째 그룹은 “결론도 안 나는데 이렇게 지루하게 할 필요가 있나”라고 생각할 수 있다. 둘 다 맞는 말이다. 다만 목적에 따라 회의하는 방식이 달라져야 한다는 것을 명심하고 회의를 진행해야 한다. 무작정 회의를 위해 모이는 것은 좋지 않다. 명확한 목적에 따라서 회의의 종류를 정해야 한다. 참고할만한 여러 가지 회의 방식은 다음과 같다.

첫 번째는 ‘토론을 위한 회의’이다. 다양한 토론을 통해 다른 사람의 의견을 계속 청취하는 것은 서로의 발전을 위해서 좋은 일이다. 그리고 이는 팀의 발전으로 연계될 수 있다. 다만 회의의 주제가 토론이 필요한 회의일 때에 한정한다. 무언가 결정이 필요한 회의에서 토론만 계속하다가는 결론도 안 나고 끝나버릴 수 있다. 이 방식의 회의를 시작

하기 전에는 반드시 미리 회의의 유형을 알려야 한다. 아니면 결론도 나지 않는데 왜 이렇게 길게 회의를 하는지 불만을 가지는 사람들이 많아질 것이기 때문이다. 이 유형의 회의를 진행할 때는 브레인 스토밍 같은 방식을 통해 최대한 자유로운 의견을 낼 수 있게 하며, 상호 비판은 가급적 하지 않는 것이 좋다. 이 회의 방식을 통해 지금까지 생각하지 못했던 창의적인 아이디어가 나올 수도 있기 때문이다.

두 번째는 ‘주요한 의사결정 회의’이다. 이 유형의 회의는 일반적으로 앞서 진행한 토론 회의에 이어 진행된다. 본 회의는 의사결정이 이루어지는 것이 최종 목적이라고 생각하고 진행해야 한다. 이미 충분한 논의를 거치지 않았는가? 여기서 주의할 점이 있다. 이 회의에서는 토론을 위한 회의에서 충분한 논의를 거친 주제를 가지고 진행하므로 이에 대한 불필요한 논의는 과감히 생략해야 한다. 토론 과정을 다시 진행하면 이 회의는 언제 끝날지 모른다. 이 회의의 목적인 최종 의사결정에 집중하고 이것을 위해서만 논의하라. 회의 종료 후에는 결정된 사안을 정리 및 요약하여 회사 구성원 모두가 알 수 있도록 공유하는 것도 잊지 말아야 한다.

세 번째는 ‘모두가 참석하는 회의’이다. 규모가 작은 회사는 직원 전체일 수도 있고, 규모가 큰 회사는 하나의 팀 단위가 될 수도 있다. 이 회의는 현재 우리 조직에서 해결하고 있는 현안, 진행하는 프로젝트 등 직원들이 관심을 가질만한 주제를 발표하는 것으로 시작한다. 발표 후에는 직원들의 의견 수렴과 질의응답 시간을 갖는다. 이 유형의 회의를 통해 직원들은 회사의 전체 상황에 대해 이해할 수 있다. 회사

는 직원들의 다양한 의견들을 직접 들어볼 수 있는 시간이 된다. 다만, 이 유형의 회의는 가급적 최소화하며 반드시 필요할 때에만 하는 것이 좋다.

마지막은 ‘스킵 레벨 미팅(Skip-Level Meeting)’이라는 회의 방식이다. 이 회의 방식은 다소 생소할 수 있다. 스타벅스는 스킵 레벨 미팅 방식의 회의를 이용한다. 매니저를 제외한 매장 직원들만 참여하는 회의를 진행하는 방식이다. 중간 관리자는 회의 진행에 부담을 줄 수 있으니 배제하고 자유로운 토론을 진행한다.

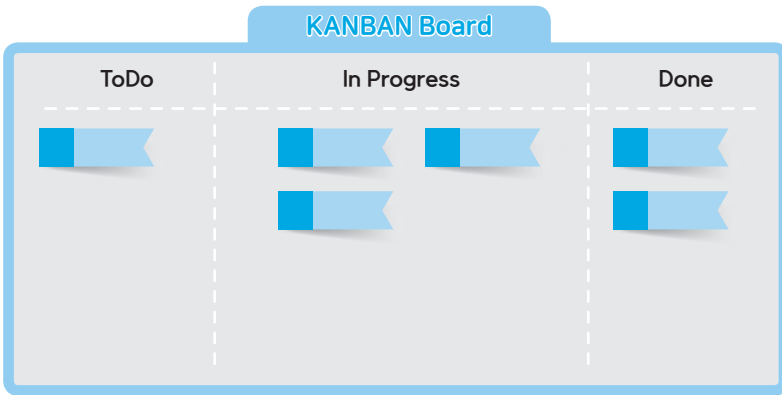
CEO와 무작위로 선정된 직원 10명이 미팅을 하는 회사들도 있다. 이런 회의를 하는 목적은 무엇일까? 중간 관리자에 대한 불만을 접수하는 시간일까? 이 회의의 궁극적인 목적은 회사의 발전이며 중간 관리자가 범하고 있는 오류들이 있으면 이러한 의견을 들어 개선할 수 있는 방향을 만드는 것이다. 연차가 낮은 직원들이 느꼈던 불편함을 경청하고 이를 기반으로 좀 더 나은 회사를 만들기 위한 회의이다. 다만, 이 회의를 자주 하는 것보다는 1년에 한두 번 정도 진행하는 것이 바람직하다.

업무 흐름을 정리하라

팀 리더가 매일 팀원들에게 “오늘은 무슨 일을 할 예정인가요?”, 혹은 “다음 주에 해야 할 일은 무엇인가요?” 등의 질문을 통해 업무 흐름을 파악한다면 어떨까? 물어보는 사람도 비효율적이지만 답변하는 사람도 피곤해진다. 이런 상황을 피하려면 업무 흐름을 명확하게 할 수

있는 도구가 필요하다.

여기서는 ‘칸반보드(Kanban Board)’라는 것을 소개한다. 칸반보드는 빠른 테스트 및 개발, 수정이 이루어지는 애자일 소프트웨어 개발 방식에 많이 활용되는 일정 관리 기법이다. 이는 아래 그림과 같이 쉽게 표현할 수 있다.



칸반보드의 예시

칸반보드는 ‘해야 할 업무(ToDo)’, ‘진행 중인 업무(In Progress)’, ‘완료된 업무(Done)’의 세 부분으로 구성된다. 각 업무는 포스트잇 1개, 회사의 모든 업무는 게시판 전체로 생각한다. 다른 색상은 다른 팀 혹은 다른 팀원을 의미한다.

활용 방법은 간단하다. 자신이 맡은 일을 포스트잇에 적어서 세 칸 중 한 곳에 붙이면 된다. 포스트잇의 색깔은 본인 고유의 색깔이 된다.

이런 방식을 활용하면 어떤 점이 좋을까? 가장 큰 장점은 현재 진행

이 잘 안 되는 부분(병목 현상)이 어디인지 한눈에 파악할 수 있다. 이에 따라서 도움이 필요한 부분을 파악하여 해결해 주기도 하고, 잘되고 있는 부분은 더욱 완성도 높게 진행할 수 있도록 격려하기도 한다. 칸반보드를 활용하면 문제의 발생을 미리 파악하고 이를 해결하기 위한 준비를 할 수 있다.

이와 같은 방법을 통해 업무를 시각화하면 누구나 전체적인 업무 진행 상황을 알 수 있게 된다. 결과적으로 제품의 생산성과 품질, 모두 향상이 가능하다.

오픈 소스 활용하기

개발과 전혀 친하지 않은 사람은 이해하지 못할 개발자들의 문화가 있다. 바로 오픈 소스이다. 직접 개발을 해보면 알겠지만 이 작업은 결코 만만한 작업이 아니다. 엄청난 시간과 노력이 있어야 하나의 결과물을 완성할 수 있다. 그런데 이렇게 완성한 결과물을 '라이브러리'라는 형태로 누구나 무료로 쓸 수 있도록 공개하고 있다. 정말 대단한 일이다.

1년 동안 만든 프로젝트를 얼굴도 모르는 사람들이 자유롭게 쓰도록 공유하기가 쉬울 것 같은가? 더욱이 무료로 공개하는 것은 더욱 어려울 것이다. 하지만 개발자들은 이러한 문화에 익숙해져 있으며 지금 이 순간도 수많은 오픈 소스들이 만들어지고 공개되고 있다.

불과 10년 전만 해도 오픈 소스의 중요성은 그렇게 크지 않았다. 내가 만든 소스 코드를 다른 사람에게 공유한다는 것 자체가 쉬운 일이

아니었다.

기업의 문화도 오픈 소스 시대에 맞춰 변화하고 있다. 기존에는 개발을 진행할 때 각각의 세션을 나누고, 각 세션의 전문가가 이를 단계별로 해결하는 방법을 활용했다. 하지만 시대가 바뀌었다. 최근에는 고객의 피드백을 바로 받아서 솔루션을 해결하고 이를 제공해야 하므로 프로세스를 한 번에 빨리 진행하는 것이 좋다. 이를 위해 필요한 것이 오픈 소스 기술이다. 여러분이 알고 있는 리눅스, 안드로이드, 데이터베이스(MySQL), 웹서버(Apache), 빅데이터 관리(Hadoop) 프로그램 등의 소프트웨어는 오픈 소스 기술을 활용한다.

최근에는 개발자가 모든 코드를 다 짜지 않는다. 이미 잘 짜여있는 오픈 소스들이 널려 있는데 다시 같은 것을 개발하며 시간을 낭비할 필요가 없지 않은가? 오픈 소스 활용을 통해 시간과 비용을 절약하는 것이 좋다. 개발을 진행하면서 좋은 오픈 소스를 찾게 되면 팀원들에게 공유하여 개발의 효율을 높이는 방법도 좋다. 또한, 우리가 만든 서비스도 오픈 소스로 공개하면 우리 회사 서비스의 성장에 도움이 될 수 있다. 물론 오픈 소스를 적극적으로 활용하기 위해서는 보안의 취약점을 잘 파악하고 관리해야 하며, 이에 대응하기 위한 기술 혹은 관련 인력을 배치하는 등의 노력이 필요하다.

- ✔ 개발자들이 일에서 의미를 찾도록 도와주는 역할을 하라.
- ✔ 개발자가 만족할 수 있는 환경(회사의 가치와 비전, 기업 문화)을 만들어라.
- ✔ 다른 기업 문화를 그대로 따라 하려고 하지 말고 우리 회사에 맞는 문화를 만드는 것이 중요하다.
- ✔ 돈을 많이 주는 것은 개발자들의 동기 유발에 큰 영향을 주지 못한다.
- ✔ 개발자들은 새로운 것을 배우고 성장해 나가는 것, 타인으로부터 인정받는 것, 본인이 만든 서비스가 사회를 위해 유용하게 쓰이는 것을 좋아하며 이는 동기 유발에 큰 영향을 줄 수 있다.
- ✔ 신입은 물론 경력 개발자도 이직 후 멘토링이 필요하며, 이는 회사적 응력을 높일 수 있는 효과적인 수단이다.
- ✔ 개발자들을 관리하는 입장이 아닌, 적절한 관심과 배려를 할 수 있는 파트너가 되어라.
- ✔ 1~2명의 우수한 직원에게 좋은 평가를 주는 것이 아닌, 팀 전체 모두가 좋은 평가를 받을 수 있도록 하라.
- ✔ 개발자의 전문성을 인정하고 이를 회사 발전을 위하여 적극 활용하라.
- ✔ 개발자들을 존중하고 이들이 원하는 방향으로 개발 및 업무를 수행할 수 있도록 도와주라.
- ✔ 개발자들이 끊임없이 배우고 새로운 시도를 할 수 있도록 지속적인 도전 기회를 주어라.
- ✔ 회사 대표는 다양한 스타일로 팀과 소통하는 시간을 주기적으로 갖는 것이 좋다.
- ✔ 상호 간 피드백 문화가 잘 정착될 수 있도록 한다.
- ✔ 전통적인 수직적 문화가 아닌 수평적 문화 정착을 위해 노력해야 한다.

- ✔ 개발자와의 신뢰 관계는 매우 중요하며 단시간에 만들어지는 것이 아니므로 꾸준한 노력이 필요하다.
- ✔ 개발자들에게 권한을 행사하려 하지 말고 이들이 자유롭게 일할 수 있도록 하라.
- ✔ 개발자와 소통할 때에는 추상적 대화는 지양하고 구체적이고 명확한 단어를 활용하여 대화하라.
- ✔ 개발팀 운영 시 개발 프로세스의 시스템화, 표준 업무 방식 제정 등 시스템 표준을 마련하는 것이 우선이다.
- ✔ 개발팀 운영 시 다른 사람의 코드를 리뷰하는 문화를 정착시켜 이들의 실수를 방지하고 효율적인 협업이 가능하도록 한다.
- ✔ 개발팀이 잘 운영되고 있는지, 문제는 없는지 등을 관심 있게 지속적으로 살피는 것이 중요하다.
- ✔ 개발팀 내 동료 문제, 업무 과중, 협업 방식 등의 문제가 발생하게 되면 현재 상황을 빠르고 효율적으로 개선할 방법을 찾아서 실천해야 한다.
- ✔ 여러 사람들의 피드백을 바탕으로 평가하는 360도 평가를 진행하되 팀 성과를 팀원 전체에게 고루 배분하는 방법이 좋다.
- ✔ 개발자들의 개인 성장이 회사의 성장으로 이어질 수 있도록 경력 개발 로드맵을 제공하여 이들이 성장할 수 있는 문화를 만들자.
- ✔ 개발자들이 다른 회사로 이직할 때에는 좋게 보내는 것이 장기적인 관점에서 큰 도움이 된다.
- ✔ 개발자들에게 많은 생각을 할 수 있는 충분한 시간과 환경을 마련해주는 것이 좋다.
- ✔ 회의를 위한 회의가 아닌 효율적인 회의를 진행하되, 업무의 특성에 따라 가장 적절한 유형의 회의 방법을 활용한다.
- ✔ 칸반보드 등을 활용하여 업무 흐름을 명확하게 정리하고 공유할 필요가 있다.
- ✔ 잘 짜여있는 오픈 소스를 적극적으로 활용하여 효율적인 개발 업무를 진행할 수 있도록 하라.



APPENDIX

부록

01 | 고용자가 알아야 할 팁과 사례들

02 | 기업에서 사용할 수 있는 형식들



01

고용자가 알아야 할 팁과 사례들

대표들이 전하는 대표들을 위한 팁

주도적인 인재를 채용하라

우리는 최대한 주도적으로 일할 수 있는 인재를 채용하려고 노력한다. 체계적으로 일을 할 수 있는 시스템이 잘 갖춰져 있으면 좋겠지만 그러기 어려운 환경이 더 많은 것이 사실이다. 이럴 때는 자기 주도적인 요소가 굉장히 중요하다. 우리는 인재들이 얼마나 자기 주도적인지, 문제 해결 능력이 얼마나 되는지를 채용에 가장 중요한 요소로 생각한다. 이를 위해 자기 주도성과 문제 해결 능력을 파악하기 위한 질문을 준비해야 한다.

초기 방향성을 확고하게 설정하라

비 ICT 기업 대표들의 인식이 변화되어야 한다고 생각한다. 대표가 개발과 기술에 관심을 가지고 회사와 어떻게 접목할 수 있을지에 대해

많은 고민을 해야 한다. 이를 통해 회사가 나아가야 하는 초기 방향성을 확고하게 설정할 필요가 있다. 이를 누가 대신해주지 않는다. 본인이 하기에 어설프더라도 직접 시작하는 것이 중요하다. '오늘의 집' 대표는 비 개발자임에도 불구하고 해외의 다양한 사례를 연구하고 고민하며 정보를 얻으려는 노력을 많이 하는 것으로 알려졌다.

명확한 역할(Role)을 제시하라

일반적으로 개발자가 IT 기업에 들어가면 자신의 역할은 명확히 보인다. 하지만 비 IT 기업에 들어가면 '어떤 일을 해야 인정받을까?', '무슨 역할을 맡는 게 좋을까?'와 같은 의문들이 생기기 마련이다. 이런 의문이 해소되지 않는다면 이 개발자는 오래 근무하지 못한다. 이를 해소하기 위해 개발자들에게 명확한 역할을 제시할 필요가 있다. 명확한 역할 제시는 업무를 효율적으로 수행할 수 있는 밑거름이 되며 지속적인 발전으로 이어진다.

대표의 스토리를 팔아서 인재를 끌고 와야 한다

스타트업은 돈을 많이 주지도 못하고 업무 여건도 좋지 못하다. 하지만 개발하는 서비스가 좋아서 열심히만 하면 대박이 날 수 있다는 희망과 구현된 서비스를 통해 사회 구성원들이 더 편리해질 수 있다는 기대 등을 할 수 있는 곳이 스타트업이다. 스타트업은 대표가 이러한 포인트를 잘 살려 인재들을 포섭해야 한다. 확실한 미션과 비전을 설정하고 발로 뛰면서 우리 기업의 스토리를 팔아야 한다.

국내외 채용 팁들과 사례

스타트업의 초창기 직원 채용 방식(ZipRecruiter)

이 회사는 스타트업의 리쿠르팅은 일반 회사의 채용과 달라야 한다고 강조했다.

첫째, 급여와 복지, 인지도 측면에서 스타트업은 절대적으로 불리하므로 구직자에게 회사의 미션과 성취할 비전을 명확히 제시해야 한다. 둘째, 창업자의 신조를 맹목적으로 믿게 하기보다는 직원이 회사의 미션과 비전을 실현하는 것에 집중할 수 있음을 보여야 한다. 셋째, 다른 직원과의 조화로운 협력이 가능한 사회적 태도가 있는지를 면접을 통해 파악해야 한다. 넷째, 채용 계약 전에 특정 프로젝트를 주어 시범적으로 일을 해보는 기간을 두는 것도 좋다고 한다.

스타트업의 인재 유치 방법(실리콘밸리)

유능한 인재 유치를 위해 가장 중요한 것은 회사의 미션과 비전을 명확히 세우는 것이다. 다양한 행사나 모임에서 회사를 알릴만한 프레젠테이션으로 브랜드 가치를 높이는 것이 중요하다. 이를 통해 구직자들의 관심을 끌게 만드는 것이다. 스타트업의 제품을 열렬히 좋아하는 팬을 직원으로 채용할 수도 있다. 인재 채용 후 성장과 성취의 기회를 제공하여 동기를 부여하는 것이 인재 영입을 위한 좋은 기업 문화를 만드는 것이기도 하다.

지원자의 핵심 가치를 검증하라(아마존)

많은 기업들이 기업의 핵심 가치나 조직문화의 적합성을 주요 선발 기준으로 사용한다고 말한다. 하지만 실제로 이를 엄격하게 적용하는 기업은 많지 않다. 아마존은 채용 시에 아마존 리더십 14개 원칙과 적합성을 중점적으로 살펴보는 전문 면접관을 두어 이 부분을 채용에 중요한 요소로 활용하고 있다. 전문 면접관을 '바 레이저(Bar Raiser)'라고 부른다. 이들은 CEO의 채용 원칙과 기준을 이해하고 채용을 주도하는 사람이다. 바 레이저는 지원자가 아무리 뛰어난 역량을 갖고 있더라도 리더십 원칙에 맞지 않는 경우 채용 거부권을 행사할 수 있다. 이들은 별도의 보상 없이 아마존에서 인정받는 인재라는 자부심을 갖고 활동한다고 한다.

진정성 있는 채용이 필요한 이유

진정성 있는 채용이란, 우리 회사의 핵심적인 모습을 그대로 진실되게 지원자들에게 보여주는 것이다. 진정성이 있는 채용은 지원자의 정신적 안정감과 회사에 대한 확신을 주어 더욱 효과적인 채용이 될 수 있다. 이러한 채용 방식은 채용 브랜딩으로 발전하여 회사의 이미지 향상에도 많은 도움을 준다. 결과적으로 유능한 개발자들이 우리 회사에 지원하게 되는 큰 요인이 될 수 있다.

진정성 있는 지원자의 경험을 끌어내는 방법

지원자의 진정성 있는 경험을 끌어내기 위해서는 여러 방법을 활용

할 수 있다. 우선 회사의 피상적인 정보가 아닌 지원자가 수행할 업무, 부서 등에 대해 구체적으로 이야기를 해주는 것이 좋다. 어떤 업무에서 무슨 서비스를 개발할지 상세하게 말해주면 지원자는 회사에 대해 더욱 신뢰할 수 있게 된다. 또한, 회사의 업무 강도와 야근 등 불편할 수도 있는 질문을 받았을 때 거짓으로 이야기하거나 회피하는 것은 좋지 않다. 솔직하게 있는 그대로를 전달해야 한다. 업무를 함께하게 될 직원들과 연계해 주는 것도 좋은 방법이다. 이들과 미리 소통하고 준비한다면 지원자의 진정성 있는 경험을 끌어내기에 충분할 것이다.

디지털 전환 성공 사례

디지털 전환 성공 사례들의 공통점은 기술에만 집착하지 않고 기존 비즈니스를 사업적 관점에서 재해석하려고 노력했다는 점이다. 몇 가지 사례를 살펴보겠다.

아마존으로 대표되는 전자 상거래가 급부상하면서 세계 최대 오프라인 유통 업체 월마트는 큰 위기를 맞았다. 하지만 월마트는 2020년 5월 이베이를 제치고 아마존에 이어 미국 온라인 유통기업 2위로 급성장했다. 그 비결은 무엇일까? 바로 오프라인 매장과 온라인 서비스를 결합한 결과이다. 월마트는 미국 전역에서 운영하는 5천여 개의 매장을 활용, 온라인 주문 후 오프라인 매장에서 픽업하는 서비스(Click&Collect 서비스)를 제공했다. 실제로 이 서비스는 가격 경쟁력이

높아서 소비자들로부터 많은 선호를 받았다. 아마존과는 차별화된 온·오프라인의 결합 서비스가 무너지가는 오프라인 유통업을 되살리는 데 큰 역할을 했다.

국내 스타트업인 정육각은 IT 기술 도입으로 오프라인 정육점의 한계를 무너뜨린 기업이다. 기존 육류 유통은 10일 이상이 걸리는데, 이러한 과정을 대폭 축소하여 소비자들이 더욱 맛있는 고기를 먹을 수 있도록 하였다. 이를 위해 AI 설비를 갖춘 자체 육류 가공 공장을 운영한다. 자체 육류 가공 공장은 전체 공정 속도를 조절하여 최적의 생산 속도를 유지할 수 있게 만들었다. 모든 과정은 데이터로 저장되며, 이는 더욱 효율적인 프로그램을 만드는 데 큰 역할을 한다. 기존 정육점이 가진 아날로그 이미지를 IT 기술을 통해 디지털로 바꿔 빠른 성장을 거듭하고 있다.

신문 비즈니스를 하는 뉴욕 타임스는 디지털 전환의 우수 사례로 꼽힌다. 2012년 말부터 뉴욕 타임스 CEO로 재직한 마크 톰슨은 취임 초기에 디지털 구독자 증가 둔화에 직면하고 있다는 사실을 알아챘다. 그는 문제 해결을 위해 콘텐츠의 질에 많은 투자를 감행했다. 더 많은 기자를 고용하고 연봉을 업계 평균 대비 3~4배 정도로 인상해 최고의 인재들을 끌어모았다. 이렇게 영입한 기자들이 만든 양질의 콘텐츠는 더 많은 독자들을 끌어모았고 이는 수익 창출로 이어졌다. 또한, 유능한 인재가 지속적으로 유입되는 선순환 구조가 되었다.

이에 그치지 않고 디지털 보고서를 작성하여 철저하게 기업 내부를 분석해 페이스북, 구글 등에서 훌륭한 엔지니어를 채용하였다. 그리고

다양한 부서에 디지털 전문가들을 고루 배치하였다. 결과적으로 디지털 구독 매출이 매 분기 10% 이상 꾸준히 증가하고 있다고 한다. 뉴욕 타임스는 디지털 전환에 성공한 대표적인 기업 중 하나로 회자되고 있다.

02

기업에서 사용할 수 있는 형식들

채용에 관한 여러 가지 형식들

막상 개발자 채용을 하려고 하면 관련 양식이나 형식들이 없어서 새롭게 만들어야 하는 경우가 많다. 그래서 이번 장에서는 개발자 채용을 위해서 활용 가능한 기본적인 프로세스와 이와 관련된 문서들을 제공한다. 또한, 회사가 구직자에게 지켜야 할, 기본적인 비즈니스 예의를 차릴 수 있는 문서(성의 있는 입사안내, 정식 입사 제안서)도 공유한다.

- 양식 다운로드할 수 있는 QR Code



- 구인 회사를 위한 채용 관련 양식들(절차순)

1. 입사 안내서
2. 구인 양식
3. 평판 조회 양식
4. 인터뷰 안내
5. 입사 제안서 양식

- 구직자(개발자)를 위한 이력서 양식

- 개발자 이력서 양식

DevRel과 관련된 프로세스

DevRel의 행사 진행 프로세스는 다음과 같다. 우선 행사를 기획하고 이에 참여할 연사를 모집하며, 참석자를 관리한다. 다음으로 행사 홍보 및 후원자를 결정하고 실제 행사를 진행하며, 진행 후에는 회계 및 평가 보고를 진행한다. 여기에서 주의할 점은 콘퍼런스나 행사를 통해 목표(목적)와 기대효과를 명확하게 해야 한다는 것이다

- 행사 준비 프로세스

1. 기획
 - 가. 기획서

- 나. 개요
- 다. 연사 초청 메일
- 2. 연사
 - 가. 연사 미팅노트
 - 나. 행사 준비 관련 정보 요청 설문
 - 다. 발표 가이드 제공
 - 라. 내부 리허설
 - 마. 발표자료 보완
 - 바. 행사장 리허설
- 3. 후원사 모집
 - 가. 후원 대상 카테고리 선정
 - 나. 후원 제안서 작성
 - 다. 후원사 모집
- 4. 홍보
 - 가. 홍보 채널 선정: SNS, 개발자 커뮤니티
 - 나. 홍보를 위한 콘텐츠 제작: 배너, 영상, 뉴스레터
 - 다. 홍보 활동 시작
- 5. 참석자
 - 가. 참석자 DB 관리
 - 나. 참석자 Q&A 수집
 - 다. 참석자 설문

6. 진행

가. 진행을 위한 큐시트 작성

나. 내부 스태프 리허설

다. 필요한 기자재 준비

7. 회계 및 평가 보고

● 행사 준비에 필요한 양식

1. 기획서
2. 연사 및 발표 자료 관리
3. 후원 제안서
4. 진행 일정
5. 큐시트

개발자는 미래 기술 발전의 핵심 동력이 될 것이며 ‘개발자가 갑’인 시대는 앞으로도 계속될 것이다.

우리 회사에서는 이러한 변화의 흐름을 빠르게 감지하고 대응할 필요가 있다.

개발자들은 일반인이 보기에 외계인 같아 보일 수 있다. ‘화성에서 온 남자, 금성에서 온 여자’라는 책처럼 서로 이해하지 못하는 부분이 많을 수 있다. 앞서 소개한 것처럼 개발자의 공유 문화, 소통 및 협업 문화, 개발 문화, 학습 문화 등 일반인이 이해하기 힘든 문화들이 많은 것이 사실이다. 하지만 이들을 이해하지 못하면 당신 기업은 절대 발전할 수 없다. 앞서 언급한 많은 것들을 잘 이해하고, 이를 기반으로 기업 문화 등 다양한 부분에 대한 준비를 해야 한다.

물론 이 가이드북에서 다루지 못한 내용도 있을 수 있다. 회사의 상황은 각자 많이 다르기 때문에 여러 변수가 발생할 수 있다. 이 가이드북의 내용과 다른 방향으로 흘러갈 수도 있다.

앞서 다룬 내용 중에 가장 중요한 것은 무엇일까? 바로 ‘개발자를

대하는 진심'이다. 개발자는 우리 가족이라는 생각으로 소중하게 대하고 깊게 이해하려고 노력한다면, 이들은 회사의 발전을 위해서 더욱 힘쓸 것이다.

이 가이드북을 통해 여러분이 4차 산업혁명의 핵심 인력인 유능한 개발자를 잘 채용하고 케어할 수 있기를 바란다.

지금까지 가이드북을 읽어주신 독자 여러분께 감사를 표한다.

GitHub



<https://github.com/innovationacademy-kr/tech-hr>

이노베이션 아카데미

서울시 강남구 개포로 416 개포디지털혁신파크 (수인분당선 개포동역 8번출구)