Employee Payroll System Smart Contract

Group Blocarc:
Caitlin Cao, Qian Zha,
Xinyi Liang, Ken Chuang

Group Blocarc Team Members

Caitlin Cao

Identifying business questions and problem statement

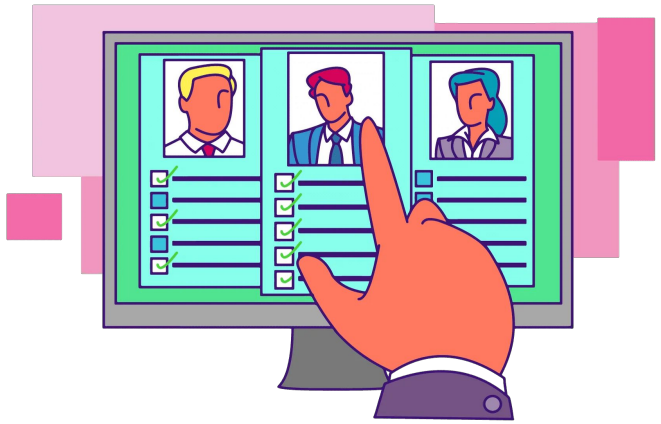
Qian Zha

Market analysis

UI design

Smart Contract Test Plan

Smart Contract Demo



Xinyi Liang

Product Idea

Business Flow

Ken Chuang

Product Idea

Today's Outline

01 **Business Plan**
Problem Statement / Traditional payroll system / Market Analysis

02 **MVP**
MVP Overview / Why blockchain technology needed / Business Flow

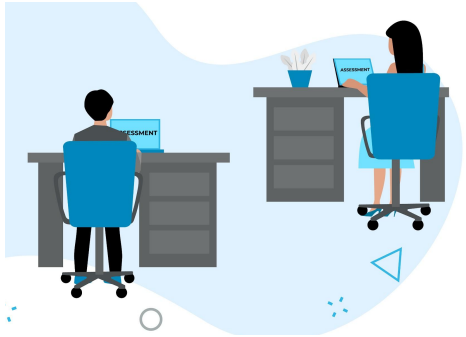
03 **Solidity Example**

04 **Minimal Single-Purpose Demo**

1.Business Plan

Problem Statement

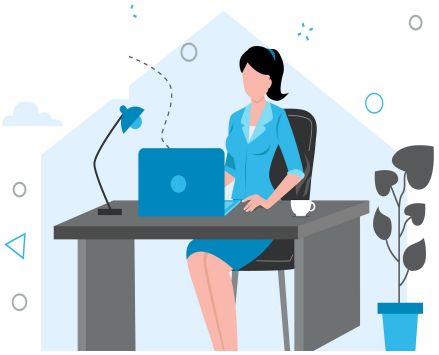
Employee Side



- Diverse demand for currency of global employees
- Suffer from incorrectness and untimeliness
- Risk of being breached, uncertainty of payment
- ……

Company Side

- Difficulty in payment settlement due to more flexible working style, location and time
- Inefficient manual payroll calculation caused errors, long hours, payment delays
- Expensive third-party outsourcing and the potential for employee and company data leakage
- ……



Looking For:
Certainty, Efficiency, Automation, Security, Convenience……

Marketing Analysis

Target Customer

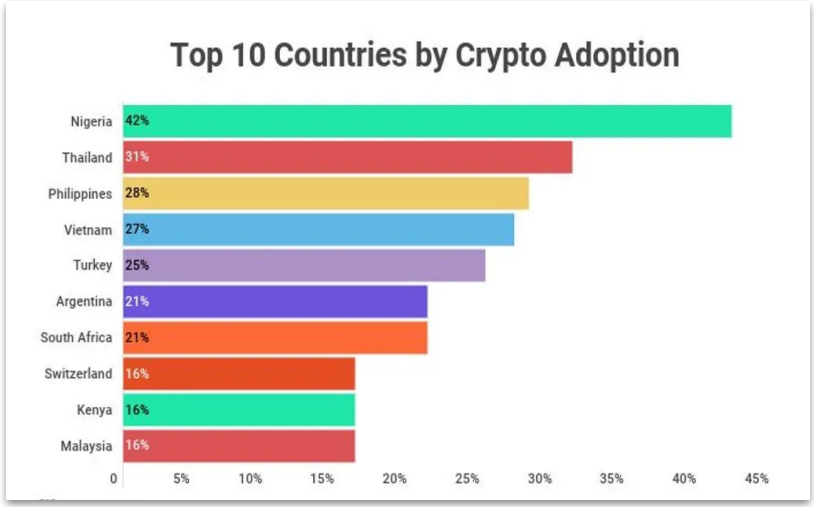
- Mid-sized and large organizations
- Either Do using software
- Or outsource to a service provider

Market Size

- Payroll software+service market was valued at **$23.55 billion** in 2021
- Estimated to reach **$55.69 billion** by 2031*

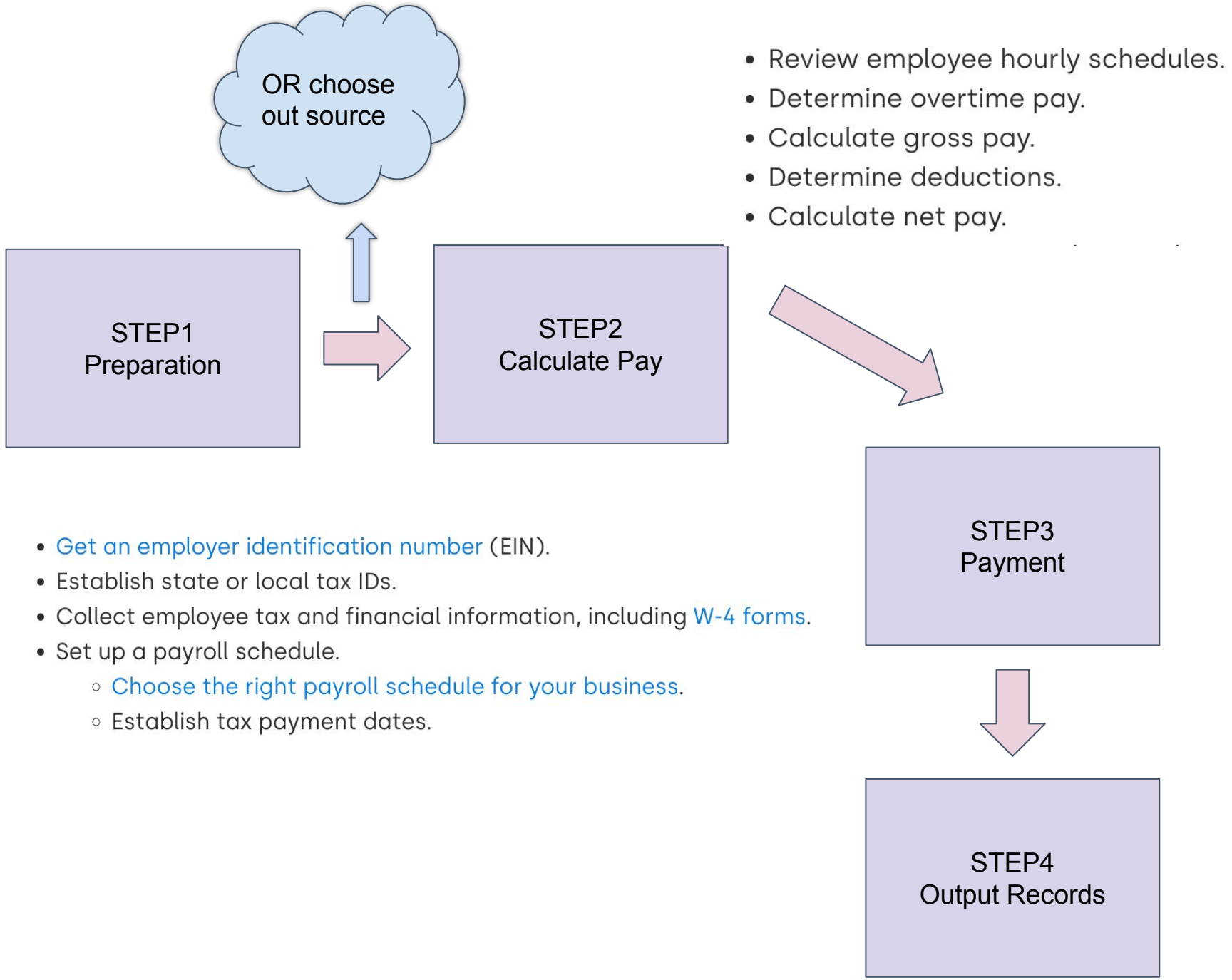
Main Market

- Adoption depends on web3 infrastructure and acceptance
- Should target marts with high web3 adoption rate



*BusinessWire 2022, Top 10 countries by crypto Adoption 2022

Traditional Payroll System Process



- Review employee hourly schedules.
- Determine overtime pay.
- Calculate gross pay.
- Determine deductions.
- Calculate net pay.

- Get an employer identification number (EIN).
- Establish state or local tax IDs.
- Collect employee tax and financial information, including W-4 forms.
- Set up a payroll schedule.
  - Choose the right payroll schedule for your business.
  - Establish tax payment dates.

2.MVP

Smart Salary Product MVP Overview

Goal

- Practicing decentralized, automatic, mandatory and accurate salary payment enable the employee to choose the currency they wish to receive
- avoid salary arrears and miscalculations

Problem Solving

The enterprise adjusts the salary payment from "active salary payment" to "authorized salary deduction"
→ can resolve issues like delayed payments or inaccurate wage calculations, especially for the client-facing service agencies or freelance independent contractors

Differentiation

No similar available product
Differentiate ourselves by providing a full-process one-stop solution system with salary payment as the core

Steps

1. Input employee information to the system
2. Use the balance in the company account to send salary to the employee
3. After the employee receive the salary they can decide which currency they would like to convert it into

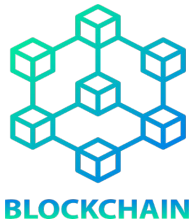
* the employers have the options to check the real-time balances of the company or any employees

Why Blockchain

Security property is one of the keys

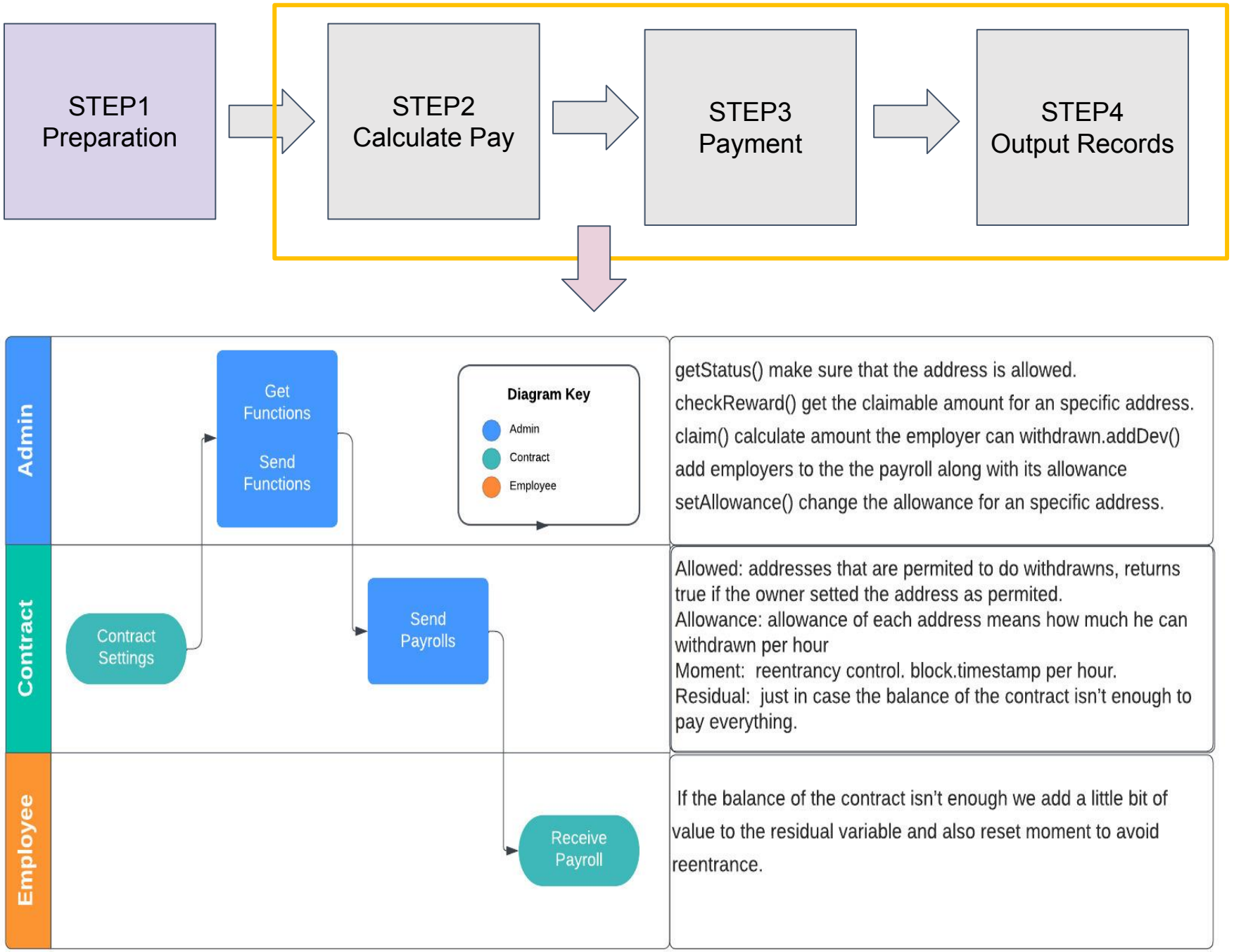
Builds trust between the company and the employees

Faster settlement compared to the traditional banking system

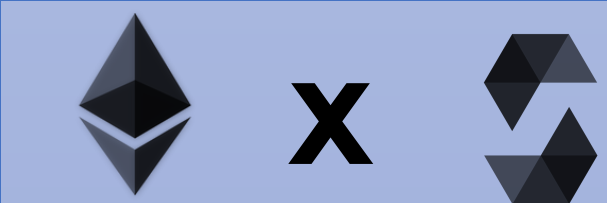


Employee can pick which kind of currency they prefer to receive

Payroll System using Smart Contract



3.Solidity Example



remix

Payroll System Smart Contract Solidity Example

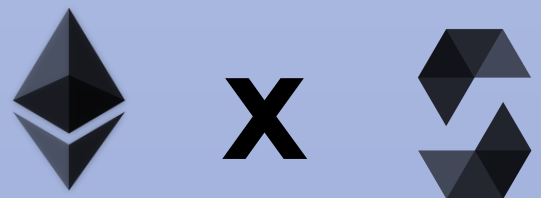
```
1   // SPDX-License-Identifier: GPL-3.0-or-later
2   pragma solidity >=0.4.4<0.9.0;
3
4   contract BinancepayrollSystem {
5
6       address public company;
7       address payable employeeAddress;
8       struct employee {
9               uint ID;            //employee ID
10              string name;        //employee name
11              string department; //employee department
12              uint salary;        //employee salary to sent
13              uint balance;       //employee account balance
14              string lastPaytime;//employee's salary last paytime
15              bool active;        //employee's status active
16              string salarytype;
17          }
18      mapping(address => employee) employeeDetails; //mapping employee address to employee information detail
19      address[] employeeList;
20
21      //payroll constructor: set the company as payroll sender
22       constructor () public {
23          company = msg.sender;
24      }
25
26      //access control modifier
27      modifier onlyCompany{
28          require (msg.sender == company,
29          "Only company owner can call this."
30          );
31          _;
32      }
```

Employee's information

mapping employee address to employee detail

set the company as the sender

access control



Payroll System Smart Contract Solidity Functions

```
function changeCompanydetail(address _companyAddress, uint balance,
                             uint newfunds,string memory lastaddtime) public {
    companyDetail[ _companyAddress] = company(balance,newfunds,lastaddtime);
}
```

change the detailed information about the company

```
function contractBalance() view public returns (uint) {
    return address(this).balance;
}
```

check the smart contract balance

```
function addEmployee(address _employeeAddress, uint ID, string memory name,
string memory department, string memory country,uint balance,uint salary,
string memory lastPaytime,bool active,uint exchangerate, uint payment) public {
    employeeDetails[ _employeeAddress] = employee(ID,name,department,country,balance,salary,
                                                  lastPaytime,active,exchangerate,payment);
    employeeList.push(_employeeAddress);
}
```

adding an employee to the company's list of employes if not already registered

```
function addFunds() public payable{
    emit deposit(msg.sender, msg.value);
}
```

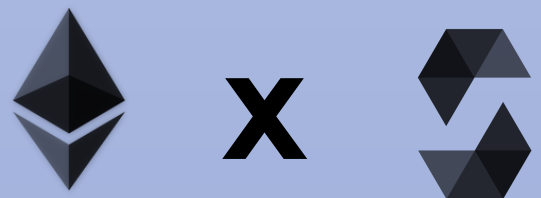
add funds to the smart contract account

```
function removeEmployee(address _employeeAddress) onlyCompany public{
    employeeDetails[ _employeeAddress].active = false;
}
```

remove an employee if the employee leaves the company

```
function getEmployeeAddress(uint _employeeDetails) public view returns (address){
    return employeeList[_employeeDetails];
}
```

get an employee's address from the company's employee list



remix

Payroll System Smart Contract Solidity Functions

```
function setEmployeesalary(address _employee, uint _salary) public payable returns (uint){
    require(_salary > 0, "salary must be greater than 0");
    return (employeeDetails[_employee].salary=_salary);
}
```

set the employee's salary based on the address

```
function getEmployeesalary(address _employee) public view returns (uint){
    return (employeeDetails[_employee].salary);
}
```

get employee's salary based on the address

```
function getEmployeeBalance(address _employee) public payable returns (uint){
    return (employeeDetails[_employee].balance=employeeDetails[_employee].salary +
            employeeDetails[_employee].balance);
}
```

get employee's new salary balance by adding the current amount to the balance in the account

```
function setEmployeepayment(address _employee, uint _balance) external payable returns (uint){
    return (employeeDetails[_employee].payment= _balance);
}
```

set the payment as the new salary balance

```
function payrollSystem(address _employee) external payable returns (uint){
    //send the payroll to the employee
    payable(companyAddress).transfer(employeeDetails[_employee].payment);
    return (employeeDetails[_employee].payment);
}
```

transfer salary to the employee

```
function etherlocal(address _employee) external payable returns (uint){
 return (employeeDetails[_employee].payment = employeeDetails[_employee].payment *
                                              employeeDetails[_employee].exchangeRate);
}
```

convert salary based on whichever country's exchange rate the user wants

4.Minimal Single-Purpose Demo

Thank you for listening!
Questions?