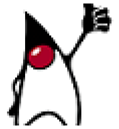
[illegible]

```scala
case class Presentation(
  title: String, author: String, date: Date, venue: String
)

Presentation(
  title = "[illegible]",
  author = "Claude-3.5 Sonnet",
  date = Date(2024, 10, 19, Sat),
  venue = "λ Kansai in Autumn 2024",
).copy(author = "kmizu")
```

□□□□

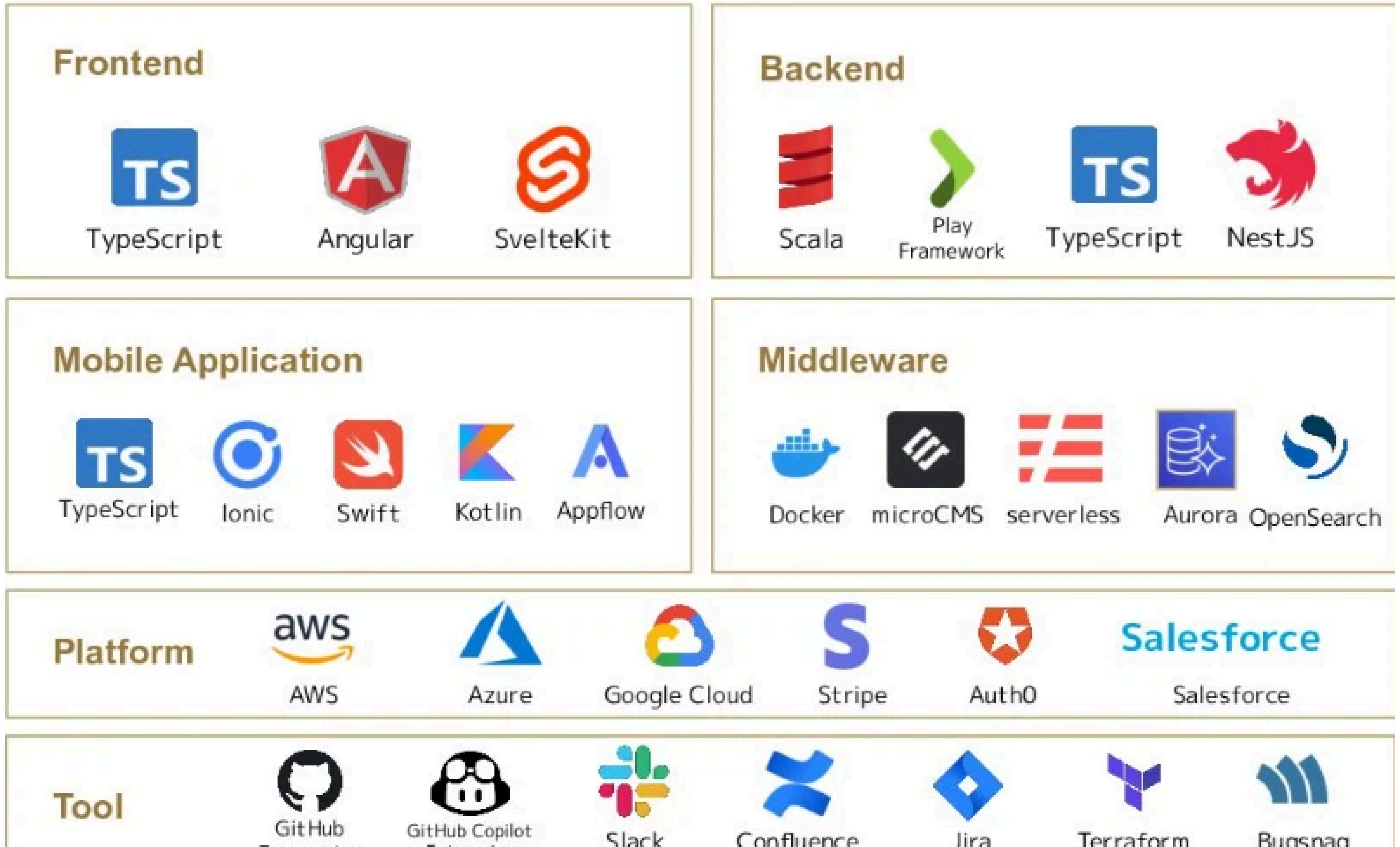


- @kmizu: https://x.com/kmizu
  - GitHub: https://github.com/kmizu
- □□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□
- Scala□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□AI□□□□□□□□□□

We are hiring!

[illegible]

- [illegible]Scala[illegible]/ TypeScript[illegible]
- [illegible]X[illegible]kmizu[illegible]DM[illegible]



[illegible]

[illegible]JavaScript[illegible]

- [illegible]
- [illegible]
- [illegible]
- [illegible]
- [illegible]
- [illegible]

[illegible]

- [illegible]
  - [illegible] <- [illegible]
- [illegible]
  - [illegible]OK

□□□ VS. □□□ in JavaScript

```javascript
// □□□
function calculateAverage(scores) {
    let total = 0;
    for (let score of scores) {
        total += score;
    }
    return total / scores.length;
}
// □□□
const calculateAverageFunctional = scores => {
    const total =
        scores.reduce((total, score) => total + score);
    return total / scores.length;
}

// □□□
const testScores = [75, 80, 90, 50, 60];
console.log(calculateAverage(testScores)); // 71
console.log(calculateAverageFunctional(testScores));  // 71
```

[illegible]

1. [illegible]
    - [illegible]
    - [illegible]
2. [illegible]
    - [illegible]
    - [illegible]
3. [illegible]
    - [illegible]
    - [illegible]

□□□□□□□□□□□□□□□ - □□□□

- □□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□□□

```
// □□□□□□
const calculateConsumptionTax = price => price * 0.1;
// □□□□□□□□□
let totalSales = 0;
const recordSale = price => {
    totalSales += price;
    return price * 1.1;
};
// □□□
console.log(calculateConsumptionTax(1000));  // □□100
console.log(recordSale(1000));  // 1100
console.log(totalSales);  // 1000
console.log(recordSale(1000));  // 1100
console.log(totalSales);  // 2000
```

□□□□□□□□□□□□□□□ - □□□

- □□□□□□□□□□□□□□□□□□□

```javascript
// □□□□□□□□□□□□□□□
const addTopping = (ramen, topping) => {
    ramen.toppings.push(topping);
    return ramen;
};
// □□□□□□□□□□□□□□
const addToppingImmutable = (ramen, topping) => ({
    ...ramen,
    toppings: [...ramen.toppings, topping]
});

// □□□
const mutableRamen = {broth: 'shoyu', toppings: ['chashu', 'menma']};
console.log(addTopping(mutableRamen, 'nori'));
// { broth: 'shoyu', toppings: ['chashu', 'menma', 'nori'] }
console.log(mutableRamen);  // □□□□□□□□□□□□□
const immutableRamen = {broth: 'miso', toppings: ['corn', 'butter']};
const newRamen = addToppingImmutable(immutableRamen, 'negi');
console.log(newRamen);
// { broth: 'miso', toppings: ['corn', 'butter', 'negi'] }
console.log(immutableRamen);  // □□□□□□□□□□□□□□
```

□□□□□□□□□□□□□□□ - □□□□

- □□□□□□□□□□□□□□□□□□□□□□□□□□□

```
const applyDiscount = (calcPrice, discount) =>
    menuItem => calcPrice(menuItem) * (1 - discount);

const regularPrice = menuItem => menuItem.price;

// □□□
const menu = [
    {name: '□□□', price: 500},
    {name: '□□', price: 550},
    {name: '□□□□', price: 700}
];

const regularCalc = regularPrice;
const discountCalc = applyDiscount(regularPrice, 0.1);  // 10%□□

menu.forEach(item =>
    console.log(`${item.name} - □□□□: ${regularCalc(item)}□, □□□□: ${discountCalc(item)}□`)
);

// □□□□□□
const prices = menu.map(regularPrice);
const expensiveItems = menu.filter(item => item.price > 600);

console.log("□□□□□□□□:", prices);
console.log("600□□□□□□□:", expensiveItems.map(item => item.name));
```

□□□□□□□□□

- □□□□□□□□□
- □□□□□□□□□□□□□□□

```
// □□□□□□□□□□□
const calculateTotalWithTax = items =>
    items.reduce((total, item) => total + item.price, 0) * 1.1;
// □□□□□□□□□□□□
let globalTaxRate = 0.1;
const calculateTotalWithDynamicTax = items => {
    const subtotal = items.reduce((total, item) => total + item.price, 0);
    return subtotal + (subtotal * globalTaxRate);
};
// □□□□□□□□□□□□□□□
const calculateTotalWithFlexibleTax = (items, taxRate) => {
    const subtotal = items.reduce((total, item) => total + item.price, 0);
    return subtotal + (subtotal * taxRate);
};
// □□□□
const testCalculateTotalWithTax = () => {
    const items = [{name: '□□□□', price: 500}, {name: '□□□□□', price: 700}];
    console.assert(calculateTotalWithTax(items) === 1320, 'calculateTotalWithTax failed');
};
const testCalculateTotalWithFlexibleTax = () => {
    const items = [{name: '□□□', price: 400}, {name: '□□□□', price: 300}];
    console.assert(calculateTotalWithFlexibleTax(items, 0.08) === 756, 'calculateTotalWithFlexibleTax failed');
};
testCalculateTotalWithTax();
testCalculateTotalWithFlexibleTax();
```

□□□□□□□□ - □□□□□□□□

```
const analyzeSales = salesData => {
    let totalSales = 0;
    let bestSellingItem = null;
    let maxQuantity = 0;

    for (let item of salesData) {
        totalSales += item.price * item.quantity;
        if (item.quantity > maxQuantity) {
            maxQuantity = item.quantity;
            bestSellingItem = item.name;
        }
    }

    const averageSales = salesData.length ? totalSales / salesData.length : 0;
    return [totalSales, averageSales, bestSellingItem];
};

// □□□
const salesData = [
    {name: '□□□', price: 500, quantity: 10},
    {name: '□□', price: 550, quantity: 15},
    {name: '□□□□', price: 700, quantity: 5}
];

console.log(analyzeSales(salesData));

// □□□□□□□□□□□□□
const analyzeSaleWithTax = (salesData, taxRate = 0.1) => {
    const [totalSales, averageSales, bestSellingItem] = analyzeSales(salesData);
    return [totalSales * (1 + taxRate), averageSales * (1 + taxRate), bestSellingItem];
};
const applyTax = (amount, taxRate = 0.1) => amount * (1 + taxRate);

console.log(analyzeSalesWithTax(salesData));
```

□□□□□□□ - □□□□□□□

```
// □□□□□□□□
const calculateTotalSales = salesData =>
    salesData.reduce((total, item) => total + item.price * item.quantity, 0);
const calculateAverageSales = salesData =>
    salesData.length ? calculateTotalSales(salesData) / salesData.length : 0;
const findBestSellingItem = salesData =>
    salesData.length ? salesData.reduce((best, item) =>
        item.quantity > best.quantity ? item : best
    ).name : null;
const analyzeSales = salesData => [
    calculateTotalSales(salesData),
    calculateAverageSales(salesData),
    findBestSellingItem(salesData)
];
// □□□
const salesData = [
    {name: '□□□', price: 500, quantity: 10},
    {name: '□□', price: 550, quantity: 15},
    {name: '□□□□', price: 700, quantity: 5}
];

console.log(analyzeSalesFunctional(salesData));

// □□□□□□□□□□□□□
const applyTax = (amount, taxRate = 0.1) => amount * (1 + taxRate);
const analyzeSalesWithTax = (salesData, taxRate = 0.1) => [
    applyTax(calculateTotalSales(salesData), taxRate),
    applyTax(calculateAverageSales(salesData), taxRate),
    findBestSellingItem(salesData)
];
console.log(analyzeSalesWithTax(salesData));
```

[illegible]

- [illegible]
  - [illegible]
- [illegible]
  - map[illegible]filter[illegible]reduce[illegible]
- [illegible]
  - Object.assign[illegible]
  - Immutable.js[illegible]
- [illegible]
  - JavaScript[illegible] by Dan Mantyla
  - [illegible] by Michał Płachta

[illegible]

- [illegible]
- [illegible]
- [illegible]
  - [illegible]
- [illegible]

[illegible]

1. [illegible]
2. [illegible]
3. [illegible]

[illegible]

[illegible]