KERJA PRAKTIK - IF184801

Perancangan dan Implementasi Permainan Petshop Idle Game menggunakan Phaser 3

Pacific Century Place Tower Lt. 26 SCBD (Sudirman Central Business District) Lot 10, Jl. Jend. Sudirman No.52-53, RT.5/RW.3, Senayan, Kec. Kby. Baru, Kota Jakarta Selatan, Daerah Khusus Ibukota Jakarta 12190

Oleh:

Geizka Wahyu Fahriza 05111840000062

Pembimbing Departemen
Abdul Munif, S.Kom., M.Sc.
Pembimbing Lapangan
Anjar Aditya Pratama

DEPARTEMEN TEKNIK INFORMATIKA
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2021

[Halaman ini sengaja dikosongkan]

KERJA PRAKTIK - IF184801

Perancangan dan Implementasi Permainan Petshop Idle Game menggunakan Phaser 3

Pacific Century Place Tower Lt. 26 SCBD (Sudirman Central Business District) Lot 10, Jl. Jend. Sudirman No.52-53, RT.5/RW.3, Senayan, Kec. Kby. Baru, Kota Jakarta Selatan, Daerah Khusus Ibukota Jakarta 12190

Oleh:

Geizka Wahyu Fahriza 05111840000062

Pembimbing Departemen
Abdul Munif, S.Kom., M.Sc.

Pembimbing Lapangan
Anjar Aditya Pratama

DEPARTEMEN TEKNIK INFORMATIKA
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2021

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN
KERJA PRAKTIK

Perancangan dan Implementasi Permainan Petshop Idle Game menggunakan Phaser 3

Oleh:

Geizka Wahyu Fahriza 05111840000062

Disetujui oleh Pembimbing Kerja Praktik:

1. Abdul Munif, S.Kom., M.Sc.
   NIP. 198608232015041004

(Pembimbing Departemen)

2. Anjar Aditya Pratama
   NIP. IDSP11083

(Pembimbing Lapangan)

[Halaman ini sengaja dikosongkan]

Perancangan dan Implementasi Permainan Petshop Idle Game menggunakan Phaser 3

Nama Mahasiswa	: Geizka Wahyu Fahriza
NRP	: 05111840000062
Departemen	: Teknik Informatika FTEIC-ITS
Pembimbing Departemen	: Abdul Munif, S.Kom., M.Sc.
Pembimbing Lapangan	: Anjar Aditya Pratama

ABSTRAK

Shopee merupakan perusahaan perdagangan elektronik. Shopee pertama diluncurkan pada tahun 2015 dan mempunyai tujuan untuk mengubah kekuatan teknologi dan mengubah dunia untuk menjadi lebih baik dengan menyediakan platform untuk menghubungkan pengguna dan pembeli dalam satu komunitas. Shopee mempunyai aplikasi yang bernama Shopee yang berbasis web. Aplikasi Shopee mempunyai fitur seperti permainan, promo, keuangan, dan lain-lain.

Permainan dibuat menggunakan framework Phaser 3 berbasis Typescript. Permainan mempunyai beberapa fitur seperti toko kucing, memberi makan kucing, dua jenis mata uang, dan passive income. Permainan dibuat untuk membuat pengguna untuk lebih sering membuka aplikasi karena permainan yang beraliran idle.

Kata Kunci : Game, Phaser 3, Idle Game

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Allah SWT atas penyertaan dan karunia-Nya sehingga penulis dapat menyelesaikan salah satu kewajiban penulis sebagai mahasiswa Departemen Teknik Informatika ITS yaitu Kerja Praktik yang berjudul: Perancangan dan Implementasi Permainan Petshop Idle Game menggunakan Phaser 3.

Penulis menyadari bahwa masih banyak kekurangan baik dalam melaksanakan kerja praktik maupun penyusunan buku laporan kerja praktik ini. Namun penulis berharap buku laporan ini dapat menambah wawasan pembaca dan dapat menjadi sumber referensi.

Melalui buku laporan ini penulis juga ingin menyampaikan rasa terima kasih kepada orang-orang yang telah membantu menyusun laporan kerja praktik baik secara langsung maupun tidak langsung antara lain:

1. Kedua orang tua penulis.
2. Bapak Abdul Munif, S.Kom., M.Sc. selaku dosen pembimbing kerja praktik sekaligus koordinator kerja praktik.
3. Bapak Anjar Aditya Pratama selaku pembimbing lapangan selama kerja praktik berlangsung.
4. Teman-teman penulis yang senantiasa memberikan semangat ketika penulis melaksanakan KP.

Surabaya, 31 Mei 2021
Geizka Wahyu Fahriza

[Halaman ini sengaja dikosongkan]

DAFTAR ISI

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

[Halaman ini sengaja dikosongkan]

DAFTAR KODE SUMBER

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

1.1. Latar Belakang

Shopee merupakan perusahaan *e-commerce* yang terbesar di Asia tenggara. Aplikasi Shopee dapat diakses melalui *desktop browser* atau *mobile*. Untuk menambah jumlah pengguna aktif Shopee, Shopee memberikan promo, voucer, dan diskon pada aplikasinya. Promo ini dapat didapatkan dengan beberapa cara, salah satunya adalah dengan memainkan permainan yang ada di Shopee. Saat ini Shopee mempunyai 4 permainan, yaitu: Shopee Candy, Shopee Bubble, Shopee Capit, dan Shopee Tanam.

Permainan Idle atau cukup juga dikenal sebagai *clicker game* atau *tap game*, merupakan permainan yang meminta pengguna untuk melakukan aktivitas yang sangat sederhana. Permainan ini biasanya akan menambahkan harta pemain, di mana ini dapat digunakan untuk meningkatkan produktivitas pemain. Dengan ini pertambahan mata uang yang terus menerus ini mengundang pemain untuk terus memainkannya.

1.2. Tujuan

Tujuan kerja praktik ini adalah menyelesaikan kewajiban nilai kerja praktik sebesar 2 SKS dan membantu PT. Shopee Indonesia untuk menyelesaikan prototipe untuk Petshop Idle Game.

1.3. Manfaat

Manfaat yang diperoleh adalah akan lebih banyak pengguna yang menggunakan aplikasi Shopee untuk bermain permainan idle tersebut. Permainan ini bertujuan

untuk mengundang pemain yang suka dengan permainan *idle* dan pengguna yang suka dengan hewan peliharaan.

1.4. Rumusan Masalah

Rumusan masalah dari kerja praktik ini adalah sebagai berikut:

1. Bagaimana arsitektur untuk permainan Petshop Idle Game?
2. Bagaimana cara mengimplementasikan permainan beraliran Idle berbasis web menggunakan Phaser 3?

1.5. Lokasi dan Waktu Kerja Praktik

Sehubungan dengan adanya pandemi dan diberlakukannya *Work From Home*, pengerjaan kerja praktik ini lakukan secara *remote*.

Adapun kerja praktik dimulai pada tanggal 25 Januari 2021 hingga 24 April 2021. Kerja praktik dilakukan pada hari Senin sampai Jumat dari pukul 09.00 hingga 18.00 dengan 1 jam istirahat.

1.6. Metodologi Kerja Praktik

Metodologi dalam pembuatan buku kerja praktik meliputi :

1.6.1. Perumusan Masalah

Untuk mengetahui kebutuhan dari Shopee, saya Bersama tim developer lainnya mendapatkan penugasan dari Pak Kevin Sutrisno mengenai permainan beraliran *idle* bertemakan hewan. Pak Kevin mengajak saya dan anggota tim berdiskusi mengenai fitur dan kebutuhan pengguna di permainan. Diskusi ini berjalan secara *asynchronous* yaitu dengan bertukar pesan. Setelah mendapatkan fitur dan kebutuhan pengguna, Pak Kevin merancang tampilan pengguna

dan menyusunnya dalam Google Slide. Saya dengan anggota tim developer lainnya berdiskusi mengenai pembagian tugas, saya mendapat tugas untuk membuat sistem toko hewan dan sistem mata uang.

1.6.2. Studi Literatur

Setelah mendapat gambaran bagaimana sistem tersebut berjalan, saya diberitahu tinjauan apa saja yang akan digunakan untuk membuat permainan di Shopee. Tinjauan yang dipakai meliputi Phaser 3, Typescript, dan Heroku. Selain itu, saya dijelaskan aturan-aturan dalam pembuatan repositori dan pengerjaan yang dilakukan menggunakan repositori Shopee.

1.6.3. Analisis dan Perancangan Sistem

Setelah tinjauan telah diberitahu, untuk merancang sistem yang baik, perlu adanya sebuah desain arsitektur sistem. Pada sistem mata uang, saya menggunakan *Singleton design pattern,* sementara untuk melakukan pertambahan uang setiap waktunya dilakukan dengan menggunakan kelas *Date* dalam Typescript serta fungsi *update* dalam Phaser 3.

1.6.4. Implementasi Sistem

Implementasi merupakan realisasi dari tahap perancangan. Pada tahap ini saya melakukan implementasi sistem permainan dan penyebaran prototipe permainan menggunakan Heroku.

1.6.5. Pengujian dan Evaluasi

Setelah permainan yang telah direncanakan telah jadi, perlu adanya evaluasi untuk menguji apakah permainan sesuai dengan fitur yang sudah ditentukan. Untuk proses pengujian dilakukan dengan menggunakan permainan yang sudah disebarkan menggunakan dan melakukan uji coba kebutuhan pengguna satu per satu. Pengujian akan dilakukan oleh Pak Kevin Sutrisno.

1.6.6. Kesimpulan dan Saran

Bab ini berisi kesimpulan dan saran yang didapatkan dari tugas selama kerja praktik.

1.7. Sistematika Laporan

1.7.1. Bab I Pendahuluan

Bab ini berisi latar belakang, tujuan, manfaat, rumusan masalah, lokasi dan waktu kerja praktik, metodologi, dan sistematika laporan.

1.7.2. Bab II Profil Perusahaan

Bab ini berisi gambaran umum PT. Shopee Indonesia mulai dari profil, lokasi perusahaan.

1.7.3. Bab III Tinjauan Pustaka

Bab ini berisi dasar teori dari teknologi yang digunakan dalam menyelesaikan proyek kerja praktik.

1.7.4. Bab IV Analisis dan Perancangan Infrastruktur Sistem

Bab ini berisi mengenai tahap analisis sistem aplikasi dalam menyelesaikan proyek kerja praktik.

1.7.5. Bab V Implementasi Sistem

Bab ini berisi uraian tahap-tahap yang dilakukan untuk proses implementasi permainan.

1.7.6. Bab VI Pengujian dan Evaluasi

Bab ini berisi hasil uji coba dan evaluasi dari aplikasi yang telah dikembangkan selama pelaksanaan kerja praktik.

1.7.7. Bab VII Kesimpulan dan Saran

Bab ini berisi kesimpulan dan saran yang didapat dari proses pelaksanaan kerja praktik.

BAB II
PROFIL PERUSAHAAN

2.1. Profil PT. Shopee Indonesia

Shopee merupakan perusahaan *e-commerce*. Shopee pertama diluncurkan pada tahun 2015. Shopee mempunyai tujuan untuk mengubah kekuatan teknologi dan mengubah dunia untuk menjadi lebih baik dengan menyediakan platform untuk menghubungkan pengguna dan pembeli dalam satu komunitas. Shopee merupakan bagian dari Sea Group yaitu perusahaan Singapura yang mempunyai bisnis berbasis internet.

2.2. Logo Perusahaan

Adapun logo perusahaan dari Shopee ditunjukkan pada Gambar 2.1.

Gambar 2.1 Logo Shopee

2.3. Visi dan Misi Perusahaan

2.3.1 Visi

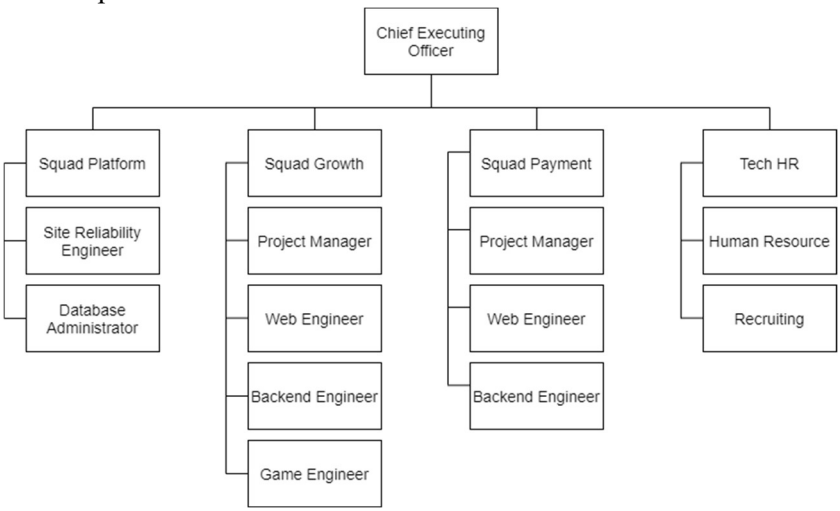
Menjadi *mobile marketplace* nomor 1 di Indonesia

2.3.2 Misi

Mengembangkan jiwa kewirausahaan bagi para penjual di Indonesia

2.4. Struktur Organisasi

Adapun struktur organisasi Shopee ditunjukkan pada Gambar 2.2.



Gambar 2.2 Struktur Organisasi Shopee

2.5. Departemen

Dalam kerja praktik ini, saya ditempatkan di departemen Game Engineer. Game Engineer bertugas untuk membuat permainan pada Aplikasi Shopee.

Permainan yang sudah dikeluarkan adalah Shopee Candy, Shopee Bubble, Shopee Capit, dan Shopee Tanam. Dalam kerja praktik ini, saya mendapatkan tugas untuk membuat prototipe permainan beraliran *idle*.

2.6. Lokasi

Pacific Century Place Tower Lt. 26 SCBD (Sudirman Central Business District) Lot 10, Jl. Jend. Sudirman No.52-53, RT.5/RW.3, Senayan, Kec. Kebayoran Baru, Kota Jakarta Selatan, Daerah Khusus Ibukota Jakarta 12190

[Halaman ini sengaja dikosongkan]

BAB III
TINJAUAN PUSTAKA

3.1. Phaser 3

Phaser adalah HTML5 framework permainan *open source* yang dikembangkan oleh Photon Storm. Phaser didesain untuk bisa berjalan di *desktop* dan *mobile* browser. Performa permainan Phaser difokuskan untuk berjalan dengan baik di *mobile browser*. Phaser menggunakan WebGL untuk *rendering*, namun akan berpindah menggunakan *Canvas* jika alat pengguna tidak mendukung WebGL [1].

Phaser digunakan dalam pengembangan permainan di Shopee. Phaser yang digunakan adalah Phaser 3. Phaser 3 digunakan karena dapat mempercepat pengembangan permainan aliran dua dimensi dalam HTML5. Phaser 3 juga mempermudah penyebaran aplikasi karena kompatibel dengan semua jenis *backend*. Ini dikarenakan kode-kode dalam Phaser 3 akan dikompilasi menjadi *script* Javascript.

3.2. Typescript

Typescript adalah bahasa pemrograman *open source* yang dibuat dengan dasar Javascript, dengan menambahkan tipe definisi statis. Tipe memberikan cara untuk mendefinisikan bentuk dari sebuah *object*, memberikan lebih banyak dokumentasi, dan memastikan kode berjalan dengan benar. Meskipun begitu, menulis tipe tidak diharuskan, karena dengan *type inference* membiarkan kamu untuk mendapatkan kekuatan tanpa menulis kode tambahan [2].

Dalam pengembangan Phaser 3, digunakan bahasa Typescript untuk mempermudah pengembangan serta

mengurangi kesalahan dalam pemrograman menggunakan fitur tipe definisi statis. Dengan ini, kode dapat menunjukkan kesalahan pada saat kompilasi dan mengurangi waktu untuk mencari kesalahan kode.

3.3. Heroku

Heroku adalah platform awan berbasis *container*. *Developers* menggunakan Heroku untuk menyebarkan, mengatur, dan mengembangkan aplikasi. Heroku sudah diatur secara penuh sehingga pengembang aplikasi dapat fokus untuk mengembangkan aplikasi tanpa bingung dalam mengatur server, perangkat keras dan infrastruktur [3].

Heroku digunakan untuk melakukan *testing* penyebaran permainan di Shopee. Heroku menyediakan *platform* yang mudah diatur dan digunakan. Dengan ini, pengembang dapat melakukan penyebaran dengan cepat untuk melakukan tes fitur tanpa mengatur *web server*.

BAB IV
ANALISIS DAN PERANCANGAN SISTEM

4.1. Analisis Sistem

Pada bab ini akan dijelaskan mengenai tahapan dalam membangun aplikasi permainan Petshop Idle Game. Hal tersebut dijelaskan ke dalam dua bagian, definisi umum aplikasi dan analisis kebutuhan.

4.1.1. Definisi Umum Aplikasi

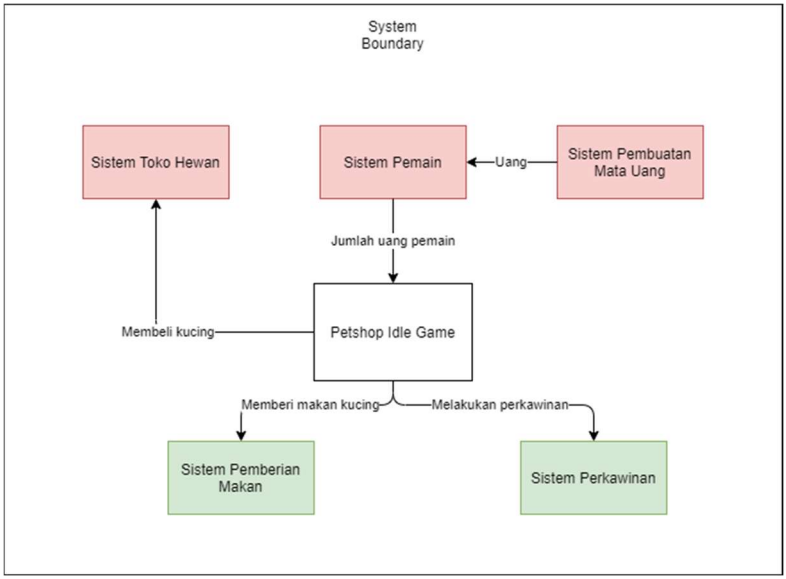
Secara umum, aplikasi permainan Petshop Idle Game merupakan permainan berbasis web di mana pengguna dapat membeli kucing dan merawat kucing tersebut. Petshop Idle Game mempunyai 2 jenis mata uang, yaitu: reguler dan premium. Mata uang ini dapat digunakan untuk membeli kucing lagi di mana nanti akan menambah koleksi atau produksi uang pengguna. Permainan mempunyai 4 fitur utama, yaitu:

a. Pemain dapat membeli kucing di toko
b. Kucing dapat membuat koin reguler atau premium
c. Pemain dapat mengembang biakan kucing
d. Pemain dapat memberi makan kucing

4.2. Perancangan Sistem

4.2.1. Desain Sistem

Desain sistem permainan Petshop Idle Game terdiri dari 4 bagian, yaitu: sistem perkawinan, sistem pemberian makan, sistem pembuatan mata uang, dan sistem toko hewan. Adapun diagram konteks sistem ditampilkan pada Gambar 4.1.

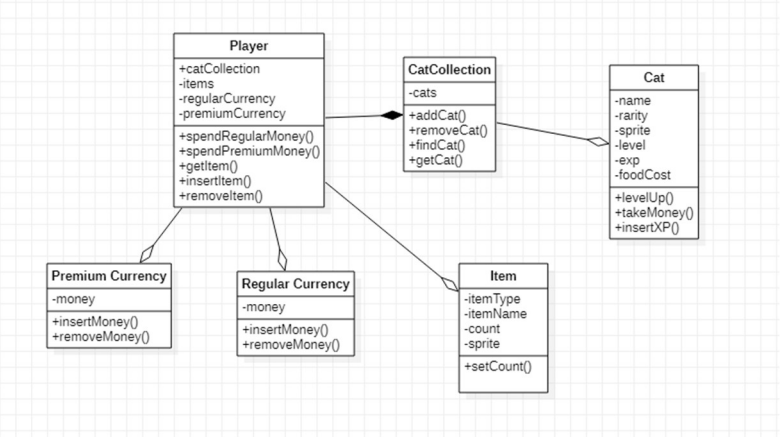


Gambar 4.1 Diagram Konteks Sistem

Pada sistem-sistem tersebut, sistem yang berwarna merah merupakan sistem yang saya kerjakan. Sistem tersebut adalah sistem toko hewan, sistem pemain, dan sistem pembuatan mata uang. Sementara sistem yang berwarna hijau merupakan sistem yang dikerjakan oleh kolega saya.

4.2.1.1. Desain Sistem Pemain

Sistem pemain mempunyai beberapa kebutuhan yang harus dipenuhi. Kebutuhan pemain adalah dapat mempunyai uang, dapat mempunyai kucing-kucing, dan dapat menyimpan item. Dengan kebutuhan tersebut, diagram kelas untuk sistem tersebut ditampilkan pada Gambar 4.2.



Gambar 4.2 Diagram Kelas untuk Sistem Pemain

Dalam desain pada Gambar 4.2, kelas `Player` mempunyai komposisi ke kelas `CatCollection` yang berfungsi untuk menyimpan kucing–kucing. Pemain juga mempunyai item, uang normal, dan uang premium.

Kelas `CatCollection` berfungsi untuk menyimpan semua objek kucing yang dipunyai oleh pemain. `CatCollection` dapat menambahkan kucing, mengurangi kucing, mencari kucing dengan nama tertentu, dan mengambil objek kucing dengan *index* tertentu.

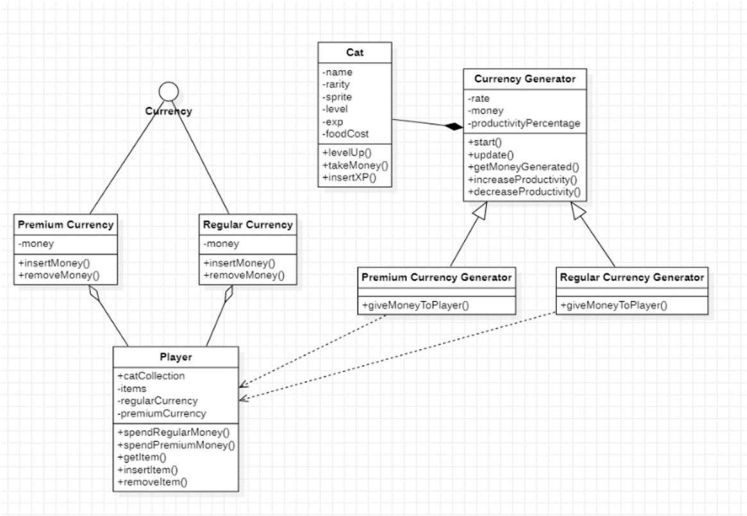
Kelas `Item` berfungsi untuk semua objek barang yang akan didapatkan oleh pemain. Di sini kelas `Item` mempunyai kebutuhan untuk mengubah jumlah item dengan jenis tersebut.

Kelas `RegularCurrency` dan `PremiumCurrency` merupakan implementasi

dari *interface* `Currency`. Semua mata uang harus mempunyai kebutuhan untuk memasukkan uang, menghilangkan uang dan melihat jumlah uang yang ada.

4.2.1.2. Desain Sistem Pembuatan Mata Uang

Sistem pembuatan uang mempunyai beberapa kebutuhan yang harus dipenuhi. Kebutuhan tersebut adalah kucing dapat membuat uang dengan *rate* tertentu dan pemain dapat mengambil uang yang dibuat oleh kucing dan menyimpannya. Diagram kelas untuk sistem pembuatan mata uang ditampilkan pada Gambar 4.3.



Gambar 4.3 Diagram Kelas untuk Sistem Pembuatan Mata Uang

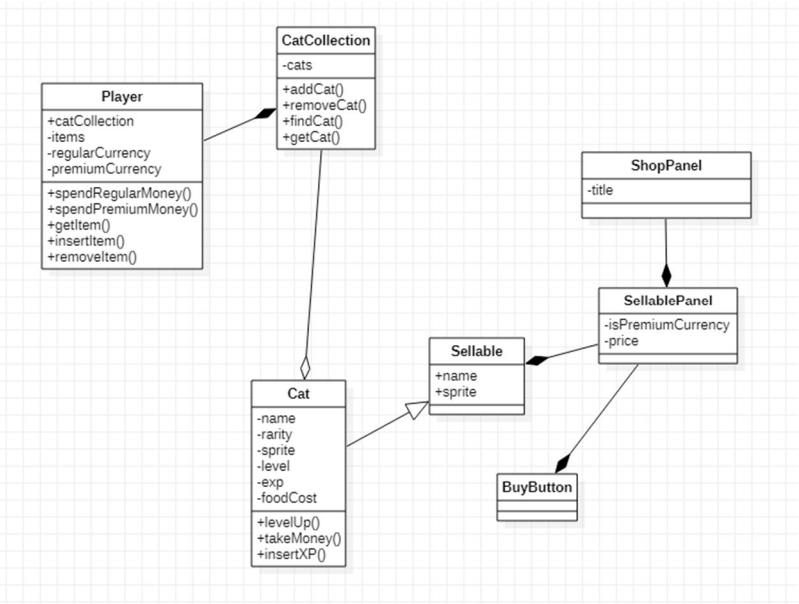
Dalam desain pada Gambar 4.3, *interface* `Currency` mempunyai implementasi `PremiumCurrency` dan `RegularCurrency`.

`Currency` mempunyai fungsi menambahkan uang, mengurangi uang, menyimpan data uang.

Setiap kelas `Cat` akan mempunyai `CurrencyGenerator`. Tipe `CurrencyGenerator` akan bergantung dengan *rarity* kucing. Kucing dengan tipe *rarity common* akan mempunyai `RegularCurrencyGenerator`, sementara kucing dengan tipe *rarity rare* akan mempunyai `PremiumCurrencyGenerator`. Setiap `CurrencyGenerator` akan mempunyai implementasi memberikan uang ke pemain melalui *singleton* `Player`.

4.2.1.3. Desain Sistem Toko Hewan

Sistem toko hewan mempunyai kebutuhan-kebutuhan seperti membeli hewan, dan menambahkan kucing yang dibeli ke dalam kepemilikan pemain. Kucing yang dibeli mempunyai *rarity* yang berbeda-beda. Kebanyakan kelas pada sistem ini nantinya akan langsung mempunyai ekstensi dari kelas *graphics* di framework Phaser 3. Diagram kelas untuk sistem toko hewan ditampilkan pada Gambar 4.4.



Gambar 4.4 Diagram Kelas untuk Sistem Toko Hewan

Dalam desain pada Gambar 4.4, `Sellable` merupakan abstraksi *interface* untuk sesuatu yang dapat dibeli dari toko. `Cat` akan mengimplementasikan `Sellable`, sehingga `Cat` dapat dijual dalam `Shop`. Setiap barang yang dapat dijual akan mempunyai panel sendiri. Di mana setiap panel, akan disimpan dalam `ShopPanel`. Setiap objek `SellablePanel` mempunyai tombol beli. Di mana setiap pembelian, kucing akan ditaruh ke `CatCollection`.

BAB V
IMPLEMENTASI SISTEM

Bab ini membahas tentang implementasi dari sistem yang saya buat. Implementasi ini akan dibagi ke dalam beberapa bagian, yaitu bagian implementasi pemain, implementasi pembuatan mata uang, implementasi toko hewan, dan penyebaran permainan.

5.1. Implementasi Pemain

Implementasi pemain berfokus untuk memenuhi kebutuhan pengguna, yaitu: menyimpan mata uang reguler, menyimpan mata uang premium, menyimpan item, dan menyimpan kucing. Adapun implementasi pemain adalah sebagai berikut:

5.1.1. Implementasi Kelas Player

Pada implementasi sistem pemain, pertama adalah membuat kelas pemain. Pemain harus memenuhi kebutuhan fungsionalitasnya. Kelas pemain ini nanti akan mempunya kelas-kelas kecil yang mempunyai tugas lebih spesifik lagi. Implementasi kelas `Player` akan ditampilkan pada Kode Sumber 5.1.

```
export default class Player {

  private static instance : Player;

  public static get Instance():Player{
    return this.instance;
  }

  private _regularCurrency : RegularCurrency;
  private _premiumCurrency : PremiumCurrency;
  private items : Item[];
  private cats : CatCollection;
```

```
  public get Cats() : CatCollection{
    return this.cats;
  }

  constructor(regularCurrency? : number, premiumCurrency? :
number){
    this._regularCurrency = new
RegularCurrency(regularCurrency);
    this._premiumCurrency = new
PremiumCurrency(premiumCurrency);
    this.items = [];
    this.cats = new CatCollection();
    Player.instance = this;
  }

  public update(){
    this.cats.update();
  }

  public insertItem(item : Item) : void{
    for(let i = 0; i < this.items.length; i++){
      if(this.items[i].Name == item.Name){
        this.items[i].Count += item.Count;
        return;
      }
    }
    this.items.push(item);
  }

  public getItemCount(_item : Item) : number{
    this.items.forEach(item => {
      if(item.Name === _item.Name){
        return item.Count;
      }
    });
    return 0;
  }

  public removeItem(_item : Item) : void {
    this.items.forEach(item => {
      if(item.Name === _item.Name){
```

```
        item.Count -= _item.Count;
      }
    });
  }

  public spendRegularMoney(money : number){
    this._regularCurrency.RemoveMoney(money);
  }

  public spendPremiumMoney(money : number){
    this._premiumCurrency.RemoveMoney(money);
  }

  public get regularCurrency():RegularCurrency{
    return this._regularCurrency;
  }

  public get premiumCurrency():PremiumCurrency{
    return this._premiumCurrency;
  }

}
```

Kode Sumber 5.1 Implementasi Kelas `Player`

5.1.2. Implementasi Penyimpanan Kucing

Untuk kebutuhan penyimpanan kucing, kelas `CatCollection` bertanggung jawab untuk menyimpan kelas `Cat`. Kelas `Cat` merupakan kelas untuk setiap objek kucing. Implementasi tersebut akan ditampilkan pada Kode Sumber 5.2 dan Kode Sumber 5.3.

```
export default class CatCollection {
```

```
  private cats : Cat[];
  private events : CatCollectionEvent[];

  public get length() : number{
    return this.cats.length;
  }

  constructor(){
    this.cats = [];
    this.events = [];
  }

  public get(index : number) : Cat{
    return this.cats[index];
  }

  public addCat(cat : Cat) : void{
    cat.isProducing = true;
    this.cats.push(cat);
    this.events.forEach(event =>{
      event.onAdd(cat);
    });
    this.events.forEach(event =>{
      event.onChange();
    });
  }

  public removeCat(cat : Cat) : void{
    let index = this.cats.indexOf(cat);
    this.cats.splice(index, 1);
    this.events.forEach(event =>{
      event.onRemove(cat);
    });
    this.events.forEach(event =>{
      event.onChange();
    });
  }

  public findCat(name : string) : Cat{
    let cat : Cat = null;
    for(let i = 0; i < this.cats.length; i++){
```

```
      if(this.cats[i].Name == name){
        cat = this.cats[i];
        break;
      }
    }
    return cat;
  }

  public update() : void{
    this.cats.forEach(cat =>{
      cat.update();
    });
  }
}
```

Kode Sumber 5.2 Implementasi Kelas `CatCollection`

```
export default class Cat implements Sellable{

  private name : string;
  private rarity : CatRarity;
  private sprite : string;
  private level: number;
  private exp: number;
  private foodCost: number;

  private currencyGenerator : CurrencyGenerator;

  constructor(name : string, rarity : CatRarity, sprite :
string, currencyGenerator : CurrencyGenerator){
    this.name = name;
    this.rarity = rarity;
    this.sprite = sprite;
    this.currencyGenerator = currencyGenerator;
    this.level = 1;
    this.exp = 0;
    this.foodCost = 100;
  }
```

```
  public clone() : Cat{
    return new Cat(this.name, this.rarity, this.sprite,
this.currencyGenerator.clone());
  }

  public get isProducing() : boolean{
    return this.currencyGenerator.isProducing;
  }

  public set isProducing(value : boolean){
    if(this.currencyGenerator.isProducing == value){
      return;
    }
    if(value){
      this.currencyGenerator.start();
    }else{
      this.currencyGenerator.stop();
    }
  }

  public update() : void{
    this.currencyGenerator.update();
  }

  public levelup(): void {
    this.level += 1;
    this.currencyGenerator.produceRate = this.NextIncome
    this.foodCost += 100
  }
  public get currencyType() : string{
    if(this.currencyGenerator instanceof
PremiumCurrencyGenerator){
      return "premium";
    }else{
      return "regular";
    }
  }

  public takeMoney() : void{
    this.currencyGenerator.giveMoneyToPlayer();
  }
```

```
  public haveMoney() : boolean{
    return this.currencyGenerator.getMoneyGenerated() > 0;
  }

  public expUp(amount: number): void {
    this.exp += amount;
    if (this.exp >= 100) {
      this.exp -= 100;
      this.levelup();
    }
  }

  public get Income(){
    return this.currencyGenerator.produceRate;
  }

  public get NextIncome(){
    return this.currencyGenerator.produceRate + 25
  }

  public get Sprite(){
    return this.sprite;
  }

  public get Name(): string {
    return this.name;
  }

  public get Rarity(): CatRarity {
    return this.rarity;
  }
}
```

Kode Sumber 5.3 Implementasi Kelas `Cat`

5.1.3. Implementasi Penyimpanan Uang

Untuk kebutuhan penyimpanan uang, kelas `RegularCurrency` bertanggung jawab untuk menyimpan mata uang reguler, dan kelas

`PremiumCurrency` bertanggung jawab untuk menyimpan mata uang premium. Implementasi kelas tersebut akan ditampilkan pada Kode Sumber 5.4 dan Kode Sumber 5.5.

```
export class RegularCurrency implements Currency {
  protected money : number;

  constructor(money?:number){
    this.money = money;
  }

  public RemoveMoney(money:number) : void{
    if(this.money < money){
      throw new Error("Money not enough");
    }
    this.money -= money;
  }

  public InsertMoney(money:number) : void {
    this.money += money;
  }

  public GetMoney() : number{
    return this.money;
  }

  public GetSpriteIconName() : string {
    return "regular_currency_icon";
  }

  public GetMoneyString(length?:number) : string {
    if(length){
      return numPadding(this.GetMoney(), length);
    }else{
      return numPadding(this.GetMoney(),
this.GetMoney.toString().length);
    }
  }
```

```
}
```

Kode Sumber 5.4 Implementasi Kelas `RegularCurrency`

```
export class PremiumCurrency implements Currency{
  protected money : number;

  constructor(money?:number){
    this.money = money;
  }

  public RemoveMoney(money:number) : void{
    if(this.money < money){
      throw new Error("Money not enough");
    }
    this.money -= money;
  }

  public InsertMoney(money:number) : void {
    this.money += money;
  }

  public GetMoney() : number{
    return this.money;
  }

  public GetSpriteIconName() : string {
    return "premium_currency_icon";
  }

  public GetMoneyString(length?:number) : string {
    if(length){
      return numPadding(this.GetMoney(), length);
    }else{
      return numPadding(this.GetMoney(),
this.GetMoney.toString().length);
    }
  }
}
```

Kode Sumber 5.5 Implementasi Kelas `PremiumCurrency`

5.1.4. Implementasi Penyimpanan Barang

Untuk kebutuhan penyimpanan barang, kelas `Item` bertugas untuk menyimpan tipe data yang didapatkan oleh pemain. Implementasi kelas tersebut akan ditampilkan pada Kode Sumber 5.6.

```
export abstract class Item implements Sellable{

  private itemType : ItemType;
  private itemName : string;
  private count : number;
  private sprite : string;

  constructor(itemType : ItemType, itemName : string,
itemSprite, count? : number){
    this.itemType = itemType;
    this.itemName = itemName;
    this.sprite = itemSprite;
    if(count){
      this.count = count;
    }else{
      this.count = 0;
    }
  }

  public get Count():number{
    return this.count;
  }

  public set Count(count : number){
    this.count = count;
  }

  public get Name():string{
    return this.itemName;
  }
```

```
  public get Type():ItemType{
    return this.itemType;
  }

  public get Sprite() : string{
    return this.sprite;
  }

}
```

Kode Sumber 5.6 Implementasi Kelas `Item`

5.2. Implementasi Pembuatan Mata Uang

Implementasi pembuatan mata uang harus memenuhi kebutuhan sebagai berikut: pembuatan mata uang harus mempunyai *rate* yang dapat ditentukan, dan pembuatan mata uang relatif ke waktu bukan banyak *update loop*. `CurrencyGenerator` ada 2 jenis, yaitu `PremiumCurrency` dan `RegularCurrency`. Adapun implementasi pemain adalah sebagai berikut:

5.2.1. Implementasi Pembuatan Mata Uang

Pada implementasi pembuatan mata uang ini, terdapat 3 kelas. Kelas `CurrencyGenerator` akan berfungsi sebagai abstraksi pembuatan mata uang. Sementara kelas `PremiumCurrencyGenerator` dan `RegularCurrencyGenerator` sebagai pembuat mata uang premium dan reguler. Kelas `CurrencyGenerator` merupakan kelas abstrak. Implementasi tersebut ditampilkan pada Kode Sumber 5.7, Kode Sumber 5.8 dan Kode Sumber 5.9.

```
export default abstract class CurrencyGenerator {
  protected rate : number;
  protected timeToGenerate : number;
```

```
  private isPause : boolean;
  private lastGetTime : number;
  private productivityPercentage : number = 1;
  protected money : number = 0;

  public get produceRate() : number{
    return this.rate;
  }

  public get produceTime() : number{
    return this.timeToGenerate;
  }

  public set produceRate(value : number){
    this.rate = value;
  }

  public set produceTime(value : number){
    this.produceTime = value;
  }

  public get isProducing() : boolean{
    return !this.isPause;
  }

  constructor(rate : number, timeToGenerate : number){
    this.rate = rate;
    this.timeToGenerate = timeToGenerate;
    this.isPause = true;
  }

  public abstract clone() : CurrencyGenerator;

  public start() : void {
    this.isPause = false;
    this.lastGetTime = new Date().getTime();
  }

  public update() : void {
    if(this.isPause)return;
    this.generateMoney();
```

```
  }

  private generateMoney() : void{
    let newTime = new Date().getTime();
    let deltaTime = newTime - this.lastGetTime;
    while(deltaTime/1000 >= this.timeToGenerate){
      this.money += this.rate * this.productivityPercentage;
      this.lastGetTime += this.timeToGenerate * 1000;
      deltaTime = newTime - this.lastGetTime;
    }
  }

  public increaseProductivity(percent : number){
    this.productivityPercentage += percent;
  }

  public decreaseProductivity(percent : number){
    this.productivityPercentage += percent;
  }

  public abstract giveMoneyToPlayer() : void;

  public getMoneyGenerated() : number{
    return this.money;
  }

  public stop() : void {
    this.isPause = true;
  }
}
```

Kode Sumber 5.7 Implementasi Kelas `CurrencyGenerator`

```
export class PremiumCurrencyGenerator extends CurrencyGenerator
{
  public clone(): CurrencyGenerator {
    return new PremiumCurrencyGenerator(this.rate,
this.timeToGenerate);
  }
```

```
  public giveMoneyToPlayer(): void {

Player.Instance.premiumCurrency.InsertMoney(this.getMoneyGenerat
ed());
    this.money = 0;
  }

}
```

Kode Sumber 5.8 Implementasi Kelas `PremiumCurrencyGenerator`

```
export class RegularCurrencyGenerator extends CurrencyGenerator
{
  public clone(): CurrencyGenerator {
    return new RegularCurrencyGenerator(this.rate,
this.timeToGenerate);
  }

  public giveMoneyToPlayer(): void {

Player.Instance.regularCurrency.InsertMoney(this.getMoneyGenerat
ed());
    this.money = 0;
  }

}
```

Kode Sumber 5.9 Implementasi Kelas `RegularCurrencyGenerator`

5.3. Implementasi Toko Hewan

Implementasi toko hewan mempunyai kebutuhan yaitu, pemain bisa membeli kucing, daftar item dapat di *scroll* saat item *overflow*, dan toko menampilkan gambar kucing yang dijual. Berikut merupakan implementasi toko hewan :

5.3.1. Implementasi Panel Toko

Pada implementasi panel toko, panel toko harus dapat menangani kondisi saat jumlah yang bisa dijual melebihi kapasitas panjang panel, dan saat pengguna menekan tempat di luar panel maka panel harus ditutup. Oleh karena itu untuk implementasi panel toko terdapat 3 kelas, yaitu `ShopPanel`, `ScrollableSellableCollection`, dan `BackgroundShopPanel`. Implementasi kelas tersebut ditampilkan di Kode sumber 5.10, Kode sumber 5.11 dan Kode sumber 5.12.

```
export default class ShopPanel extends
Phaser.GameObjects.Container{

  private shopPanel : Phaser.GameObjects.Image;
  private title : Phaser.GameObjects.Text;

  private scrollableCollection : ScrollingSellableCollection;

  constructor(scene : Phaser.Scene, x : number, y : number){
    super(scene, x, y);
    this.depth = 999;

    this.add(new BackgroundShopPanel(scene, this));

    this.shopPanel = new Phaser.GameObjects.Image(scene, 0, 0,
"shop_panel");
    this.shopPanel.displayWidth = 600;
    this.shopPanel.displayHeight = 500;
    this.shopPanel.depth = 1000;
    this.shopPanel.setInteractive();

    scene.input.setDraggable(this.shopPanel);
    this.add(this.shopPanel);
    // this.scene.add.existing(this.shopPanel);
```

```
    this.title = new Phaser.GameObjects.Text(this.scene, 0, -
this.shopPanel.displayHeight/2 + 50, "SHOP", {fontSize:"60px",
fontFamily:'Trebuchet MS'});
    this.title.setOrigin(0.5, 0);
    this.add(this.title);

    this.scrollableCollection = new
ScrollingSellableCollection(scene, 0,
this.shopPanel.displayHeight/2 - 353/2 - 30, 488, 353);
    this.add(this.scrollableCollection);
    this.scrollableCollection.addMasking();

    this.shopPanel.on("drag", (pointer : any, dragX : number)=>{
      // console.log(dragX - this.x);
      this.scrollableCollection.shiftPanel(dragX);
    })

    this.shopPanel.on("dragend", (pointer : any, dragX :
number)=>{
      this.scrollableCollection.resetDrag();
    })

    this.scene.add.existing(this);
    this.hide();
  }

  public addItem(item : Item, price : number,
isPremiumCurrency? : boolean){
    let itemPanel : SellablePanel =
this.scrollableCollection.addSellablePanel(new
ItemPanel(this.scene, 0, 0, item));
    itemPanel.setPrice(price);
    if(isPremiumCurrency != null){
      itemPanel.setIsPremiumCurrency(isPremiumCurrency);
    }
  }

  public addCat(cat : Cat, price : number, isPremiumCurrency? :
boolean){
    let itemPanel : SellablePanel =
this.scrollableCollection.addSellablePanel(new
```

```
CatPanel(this.scene, 0, 0, cat));
    itemPanel.setPrice(price);
    if(isPremiumCurrency != null){
      itemPanel.setIsPremiumCurrency(isPremiumCurrency);
    }
  }

  public update() : void{
    this.shopPanel.visible = this.visible;
    this.scrollableCollection.update();
  }

  public show() : void{
    this.visible = true;
  }

  public hide() : void{
    this.visible = false;
  }

  public toggle() : void{
    if(this.visible){
      this.hide();
    }else{
      this.show();
    }
  }

}
```

Kode Sumber 5.10 Implementasi Kelas `ShopPanel`

```
class ScrollingSellableCollection extends
Phaser.GameObjects.Container {

  private sellablePanels : SellablePanel[];
  private drag : number = 0;
  private scroll : number = 0;
  private padding : number = 20;
  private totalItemWidth : number = 0;
```

```
  private leftMostPosition : number = 0;
  private rightMostPosition : number = 0;

  private graphicMask : Phaser.GameObjects.Graphics;

  private boundary : Phaser.Geom.Rectangle;

  private itemPosition : number = 0;

  constructor(scene : Phaser.Scene, x : number, y : number,
width : number, height : number){
    super(scene, x, y);
    this.width = width;
    this.height = height;
    this.sellablePanels = [];
    this.leftMostPosition = -width/2 + this.padding;
    this.rightMostPosition = width/2 - this.padding;
  }

  public addMasking(){
    this.boundary = new
Phaser.Geom.Rectangle(this.parentContainer.x + this.x -
this.width/2 + 10, this.parentContainer.y + this.y -
this.height/2 + 10, this.width - 20, this.height - 20)
    this.graphicMask = new
Phaser.GameObjects.Graphics(this.scene);
    this.graphicMask.fillRoundedRect(this.boundary.x,
this.boundary.y, this.boundary.width, this.boundary.height, 40);
    // this.scene.add.existing(this.graphicsMask);
    this.mask = new
Phaser.Display.Masks.GeometryMask(this.scene, this.graphicMask);
  }

  public update() : void{
    this.calculatePosition();
    this.constraintScroll();
    this.disableInvisibleButtons();
    this.updateItemPanelPosition();
  }

  private isItemEnoughForScrolling() : boolean{
```

```
    return this.totalItemWidth < this.rightMostPosition -
this.leftMostPosition;
  }

  private disableInvisibleButtons() : void{
    this.sellablePanels.forEach(panel =>{
      panel.checkBuyButton(this.boundary);
    });
  }

  private calculatePosition() : void{
    this.itemPosition = this.leftMostPosition + this.scroll;
  }

  private constraintScroll() : void{
    if(this.isItemEnoughForScrolling())return;
    if(this.itemPosition > this.leftMostPosition){
      this.scroll = 0;
    }else if(this.itemPosition + this.totalItemWidth <
this.rightMostPosition){
      this.scroll = -(this.totalItemWidth -
(this.rightMostPosition-this.leftMostPosition));
    }
    this.calculatePosition();
  }

  private updateItemPanelPosition() : void{
    for(let i = 0; i < this.sellablePanels.length; i++){
      let width = this.sellablePanels[i].getBounds().width;
      this.sellablePanels[i].x = this.itemPosition + (i+0.5) *
width + i * this.padding;
    }
  }

  public shiftPanel(drag : number){
    if(this.isItemEnoughForScrolling())return;
    this.scroll -= this.drag - drag;
    this.drag = drag;
  }

  public resetDrag() : void{
```

```
    this.drag = 0;
  }

  public addSellablePanel(sellablePanel : SellablePanel) :
SellablePanel{
    this.sellablePanels.push(sellablePanel);
    this.add(sellablePanel);
    let width = sellablePanel.getBounds().width;
    if(this.sellablePanels.length == 1){
      this.totalItemWidth += width;
    }else{
      this.totalItemWidth += width + this.padding;
    }
    return sellablePanel;
  }

  public get SellablePanels() : SellablePanel[]{
    return this.sellablePanels;
  }

}
```

Kode Sumber 5.11 Implementasi Kelas `ScrollableSellableCollection`

```
class BackgroundShopPanel extends Phaser.GameObjects.Graphics {

  private width : number;
  private height : number;

  constructor(scene : Phaser.Scene, shopPanel : ShopPanel){
    super(scene);
    this.width = scene.game.scale.gameSize.width;
    this.height = scene.game.scale.gameSize.height;
    this.fillStyle(0x000000, 0.2);
    this.depth = 1;
    this.fillRect(-this.width/2, -this.height/2, this.width,
this.height);
    this.setInteractive(new Phaser.Geom.Rectangle(-this.width/2,
-this.height/2, this.width, this.height),
this.hitCallback).on("pointerdown", ()=>{
```

```
      shopPanel.hide();
    });
  }

  private hitCallback(hitArea, x, y, gameObject){
    let rect : Phaser.Geom.Rectangle = hitArea as
Phaser.Geom.Rectangle;
    return rect.contains(x, y);
  }

}
```

Kode Sumber 5.12 Implementasi Kelas `BackgroudShopPanel`

5.3.2 Implementasi Panel Barang di Panel Toko

Implementasi panel barang harus memenuhi kebutuhan dapat menampikan gambar barang dan nama barang. Jenis panel barang juga ada 2, yaitu panel barang dan panel kucing. Implementasi panel barang terdapat 3 kelas, yaitu kelas *abstract* `SellablePanel`, `ItemPanel`, dan `CatPanel`. Implementasi kelas tersebut ditampilkan di Kode Sumber 5.13, Kode Sumber 5.14, dan Kode Sumber 5.15.

```
export default abstract class SellablePanel extends
Phaser.GameObjects.Container{

  private panelImage : Phaser.GameObjects.Image;
  private itemImage : Phaser.GameObjects.Image;
  private buyButton : BuyButton;
  private itemText : Phaser.GameObjects.Text;
  private itemPrice : Phaser.GameObjects.Text;
  private price : number;
  protected item : Sellable; // Can be cat or item
  private isPremiumCurrency : boolean = false;
  private boundary : Phaser.Geom.Rectangle;

  public get worldPosX() : number{
```

```
    return this.parentContainer.x + this.x;
  }

  public get worldPosY() : number{
    return this.parentContainer.x + this.x;
  }

  constructor(scene : Phaser.Scene, x : number, y : number,
item : Sellable){
    super(scene, x, y);
    this.initComponent();
    this.setItem(item);
  }

  private initComponent() : void{
    this.depth = 1;
    this.panelImage = new Phaser.GameObjects.Image(this.scene,
0, 0, "item_panel");
    this.itemImage = new Phaser.GameObjects.Image(this.scene, 0,
-80, "item_panel");
    this.itemText = new Phaser.GameObjects.Text(this.scene, 0,
0, "item", {fontSize:"20px", fontFamily:"Trebuchet MS"});
    this.itemPrice = new Phaser.GameObjects.Text(this.scene, 0,
50, "0", {fontSize:"18px", fontFamily:"Trebuchet MS"});
    this.itemPrice.setOrigin(0.5);
    this.itemText.setOrigin(0.5);
    this.panelImage.displayWidth = 195.2;
    this.panelImage.displayHeight = 317.7;
    this.buyButton = new BuyButton(this.scene, 0, 100, this);
    this.add([this.panelImage, this.buyButton, this.itemImage,
this.itemText, this.itemPrice]);
    this.scene.add.existing(this);
  }

  //disable interactivity when not visible in mask
  public checkBuyButton(bound : Phaser.Geom.Rectangle){

this.buyButton.setClickable(this.buyButton.isVisible(bound));
    // if(this.buyButton.isVisible(bound)){
    // }else{
    //   this.buyButton.disableInteractive();
```

```
    // }
  }

  private setItem(item : Sellable) : void{
    this.item = item;
    this.itemText.text = item.Name;
    this.itemImage.setTexture(item.Sprite);
    this.itemImage.displayWidth = 125;
    this.itemImage.displayHeight = 125;
  }

  public setPrice(price : number) : void{
    this.itemPrice.text = "" + price;
    this.price = price;
  }

  public buy() : void{
    try{
      if(this.isPremiumCurrency){
        Player.Instance.spendPremiumMoney(this.price);
      }else{
        Player.Instance.spendRegularMoney(this.price);
      }
    }catch(error){
      return;
    }
    this.giveItemToPlayer();
  }

  protected abstract giveItemToPlayer() : void;

  public setIsPremiumCurrency(value : boolean){
    this.isPremiumCurrency = value;
  }

}
```

Kode Sumber 5.13 Implementasi Kelas `SellablePanel`

```
export class ItemPanel extends SellablePanel {
```

```
  constructor(scene : Phaser.Scene, x : number, y : number,
item : Item){
    super(scene, x, y, item);
  }

  protected giveItemToPlayer(): void {
    Player.Instance.insertItem((this.item as Item));
  }

}
```

Kode Sumber 5.14 Implementasi Kelas `ItemPanel`

```
export class CatPanel extends SellablePanel {

  constructor(scene : Phaser.Scene, x : number, y : number,
cat : Cat){
    super(scene, x, y, cat);
  }

  protected giveItemToPlayer(): void {
    Player.Instance.Cats.addCat((this.item as Cat).clone());
  }

}
```

Kode Sumber 5.15 Implementasi Kelas `CatPanel`

5.4. Penyebaran Permainan

Setelah implementasi, kita menyebarkan permainan ke internet menggunakan Heroku. Berikut merupakan langkah penyebaran permainan.

5.4.1. Mengubah Konfigurasi Node JS

Sebelum permainan dapat disebarkan menggunakan Heroku, pertama konfigurasi bawaan harus

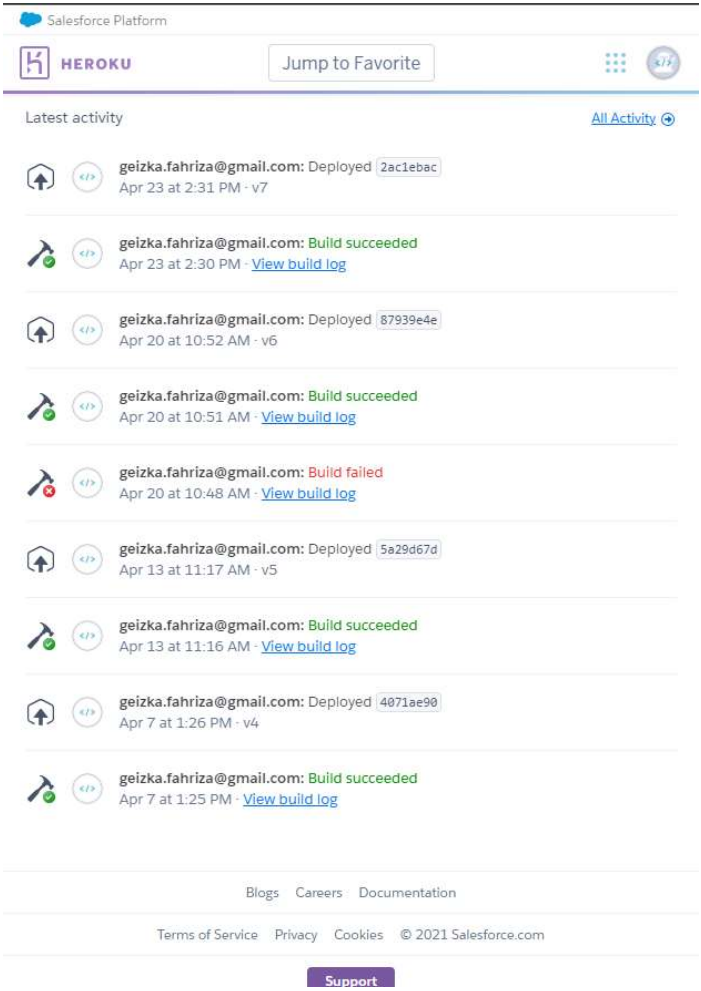
diubah agar sesuai dengan konfigurasi Heroku. Konfigurasi untuk penyebaran ditampilkan di Kode Sumber 5.16.

```
{
  "name": "phaser-game",
  "version": "0.1.0",
  "description": "starting scene",
  "main": "app.js",
  "scripts": {
    "build": "webpack",
    "start": "node app.js",
    "dev": "webpack --watch & live-server --port=8085"
  },
  "author": "kevin sutrisno",
  "license": "MIT",
  "devDependencies": {
    "live-server": "^1.2.1",
    "phaser": "^3.22.0",
    "ts-loader": "^6.2.1",
    "typescript": "^3.7.4",
    "webpack": "^4.41.5",
    "webpack-cli": "^3.3.10"
  },
  "dependencies": {
    "express": "^4.17.1"
  }
}
```

Kode Sumber 5.16 Konfigurasi Penyebaran

5.4.2. Penyebaran menggunakan Heroku

Penyebaran dilakukan dengan menggunakan Heroku CLI. Setelah melakukan instalasi program Heroku CLI, kita menambahkan folder permainan ke dalam *repository* Heroku. Setelah menambahkan permainan, heroku membuat *url*. Tampilan Heroku setelah penyebaran permainan ditunjukkan oleh Gambar 5.1.



Gambar 5.1 Tampilan Heroku setelah Penyebaran Permainan

BAB VI
PENGUJIAN DAN EVALUASI

Bab ini menjelaskan tahap uji coba terhadap Permainan Petshop Idle Game. Pengujian dilakukan untuk memastikan fungsionalitas dan kesesuaian hasil implementasi sistem dengan analisis dan perancangan sistem.

6.1. Tujuan Pengujian

Pengujian dilakukan terhadap Permainan Petshop Idle Game guna menguji apakah semua kebutuhan pengguna terpenuhi.

6.2. Batasan Pengujian

Prototipe permainan mempunyai batasan masalah yang tidak harus ditangani. Adapun batasan masalah adalah sebagai berikut:

a. Pengguna tidak mempunyai batasan jumlah kucing yang dimiliki
b. Kucing tidak mempunyai batasan jumlah maksimum koin yang dibuat
c. Koin reguler dan premium tidak mempunyai batasan jumlah koin
d. Pengguna dapat membeli kucing yang sama secara terus menerus di toko

6.3. Kriteria Pengujian

Penilaian atas pencapaian tujuan pengujian didapatkan dengan memperhatikan apakah permainan dapat memenuhi kebutuhan berikut:

a. Pengguna dapat mempunyai hewan peliharaan kucing

b. Pengguna mempunyai dua jenis uang, yaitu reguler dan premium
c. Kucing *common* dapat membuat uang dengan tipe reguler
d. Kucing *rare* dapat membuat uang dengan tipe premium
e. Pengguna dapat membeli kucing baru
f. Panel toko dapat di-*scroll* jika jumlah barang yang dijual lebih dari panjang panel
g. Panel toko dapat menampilkan daftar item yang dapat dibeli

6.4. Skenario Pengujian

Skenario pengujian dilakukan dengan melakukan peran sebagai pengguna yang akan menjalankan fitur-fitur. Pengujian yang telah dilakukan adalah sebagai berikut:

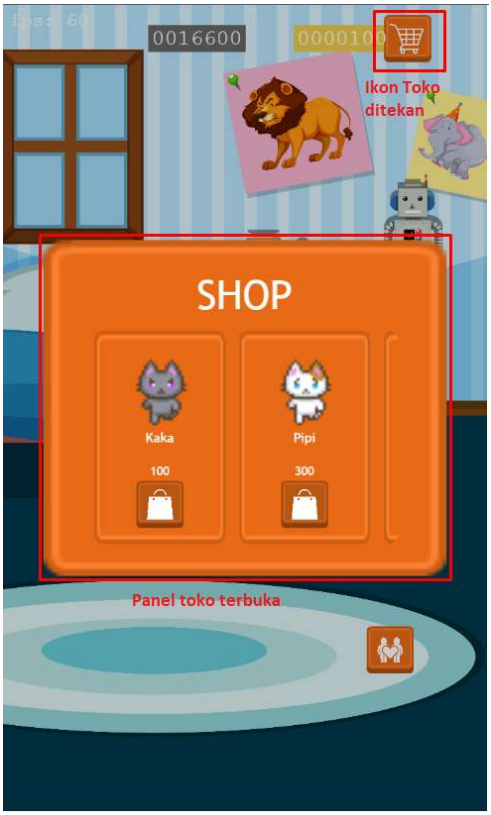
1. Sesuai dengan Gambar 6.1, Pemain dapat membuka permainan di browser, pemain mempunyai dua jenis koin, yaitu koin reguler yang berwarna perak dan koin premium yang berwarna emas. Pemain juga mempunyai dua kucing *common*.

Gambar 6.1 Pemain membuka permainan di browser

2. Sesuai dengan Gambar 6.2, Pemain menekan ikon koin yang ada di atas kucing untuk mengambil koin yang sudah dikumpulkan oleh kucing. Terlihat pada Gambar 6.2 bahwa jumlah koin pengguna bertambah menjadi 16600.

Gambar 6.2 Pemain mengambil koin reguler pada kucing

3. Pemain membuka panel toko. Seperti yang terlihat pada Gambar 6.3 Pemain dapat membuka toko dengan menekan ikon toko di pojok kanan atas. Panel toko juga dapat di-*scroll* ke kiri atau ke kanan.



Gambar 6.3 Pemain membuka toko

4. Pemain membeli kucing dengan tipe *rare*. Pemain membeli kucing *rare* di toko. Pada Gambar 6.4, terlihat bahwa pemain sudah membeli dua kucing *rare* yang menghasilkan koin premium. Setelah membeli, koin reguler pemain juga berkurang menjadi 16200.

Gambar 6.4 Pemain membeli kucing *rare*

5. Pemain mengambil koin premium dari kucing tipe *rare*. Terlihat pada Gambar 6.5, pemain sudah mengambil koin premium yang dihasilkan oleh kucing *rare* dan koin premium pemain sudah bertambah menjadi 360.

Gambar 6.5 Pemain mengambil koin premium

6.5. Evaluasi Pengujian

Hasil pengujian dilakukan terhadap pengamatan mengenai perilaku Permainan Petshop Idle Game terhadap kasus skenario uji coba. Tabel 6.1 menjelaskan hasil uji coba terhadap aplikasi yang telah dibuat.

Tabel 6.1 Hasil Evaluasi Pengujian

Kriteria Pengujian	**Hasil Pengujian**
Pengguna dapat mempunyai hewan peliharaan kucing	Terpenuhi
Pengguna mempunyai dua jenis uang, yaitu reguler dan premium	Terpenuhi
Kucing *common* dapat membuat uang dengan tipe reguler	Terpenuhi
Kucing *rare* dapat membuat uang dengan tipe premium	Terpenuhi
Pengguna dapat membeli kucing baru	Terpenuhi
Panel toko dapat menampilkan daftar item yang dapat dibeli	Terpenuhi
Panel toko dapat di-*scroll* jika jumlah barang yang dijual lebih dari panjang panel	Terpenuhi

BAB VII
KESIMPULAN DAN SARAN

7.1. Kesimpulan

Kesimpulan yang didapat setelah melakukan perancangan dan implementasi permainan Petshop Idle Game menggunakan Phaser 3 pada kegiatan kerja praktik di PT. Shopee Indonesia adalah sebagai berikut:

a. Arsitektur permainan yang dibangun telah sesuai dengan permintaan.
b. Dengan adanya aplikasi Permainan Petshop Idle Game, Shopee dapat menambahkan permainan tersebut untuk mendapatkan pengguna yang suka kucing atau suka bermain permainan beraliran Idle.

7.2. Saran

Saran untuk perancangan dan implementasi permainan Petshop Idle Game adalah sebagai berikut:

a. Untuk implementasi generasi uang, sebaiknya menyimpan *session* agar dapat kucing tetap melakukan pembuatan uang selama pengguna tidak membuka web
b. Sebaiknya terdapat maksimum jumlah kucing yang dapat dimiliki oleh seorang pemain agar tidak terjadi *overflow*.
c. Untuk *sprite* dibuat agar sesuai dengan *pixel size* dalam Phaser 3 karena Phaser 3 masih belum mendukung *9-slice scaling*.

[Halaman ini sengaja dikosongkan]

DAFTAR PUSTAKA

[1] R. Davey, “How to Learn the Phaser HTML5 Game Engine,” 12 December 2013. [Online]. Available: https://gamedevelopment.tutsplus.com/articles/how-to-learn-the-phaser-html5-game-engine--gamedev-13643.

[2] “TypeScript,” [Online]. Available: https://www.typescriptlang.org/. [Diakses 4 June 2021].

[3] “Heroku,” Heroku, [Online]. Available: https://www.heroku.com/about. [Diakses 4 June 2021].

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS

Nama	: Geizka Wahyu Fahriza
Tempat, Tanggal Lahir	: Jombang, 29 Agustus 2001
Jenis Kelamin	: Laki - Laki
Agama	: Islam
Telepon	: +6281314512001
Email	: geizka.fahriza@gmail.com

PENDIDIKAN FORMAL

2018 – sekarang	: S1 Teknik Informatika ITS
2015 – 2018	: SMA Negeri 2 Jombang
2013 – 2015	: SMP Negeri 2 Jombang
2012 – 2013	: SD Negeri Jombatan 3
2008 – 2012	: SD Islam Athirah Bukit Baruga
2007 – 2008	: SD Negeri Purwantoro 2 Malang

KEMAMPUAN

- *Backend Programming* (Node JS)
- *Game Programming* (Unity, Phaser 3)
- *Programming* (C, C#, Java)
- *Database Management* (MySQL)

AKADEMIS

Kuliah	: Departemen Teknik Informatika – Fakultas Teknologi Elektro dan Informatika Cerdas, ITS
Angkatan	: 2018
Semester	: 6 (Enam)