## 中国矿业大学计算机学院

**19 级本科生课程设计报告**

课程名称 程序设计综合实践

报告时间 2021.1.13

学生姓名 吕迎朝

学 号 08193109

专 业 数据科学与大数据技术

任课教师 李鸣

## 成绩考核

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 编号 | 课程教学目标 | | | 占比 | | 得分 |
| 1 | **目标1：**掌握一门计算机高级语言，并能使用特定的软件开发工具，设计、开发、调试及运行应用程序。 | | | 20% | |  |
| 2 | **目标2：**针对具体的应用问题，进行功能需求分析，确定设计目标，并能绘制算法流程图。 | | | 20% | |  |
| 3 | **目标3：**在进行需求分析的基础上，设计软件运行界面、关键类、编写代码，调试并正确运行满足需求的应用程序。 | | | 60% | |  |
| 总成绩 | | | | | |  |
| 指导教师 | |  | 评阅日期 | |  | |

**实验一 简单计算器**

1 系统概述

* 该系统是由Flutter+Flask+Mysql实现的拼图游戏。可以自定义图片和难度，可以实现登录，游戏积分排行榜，游戏最快用时，游戏计时。

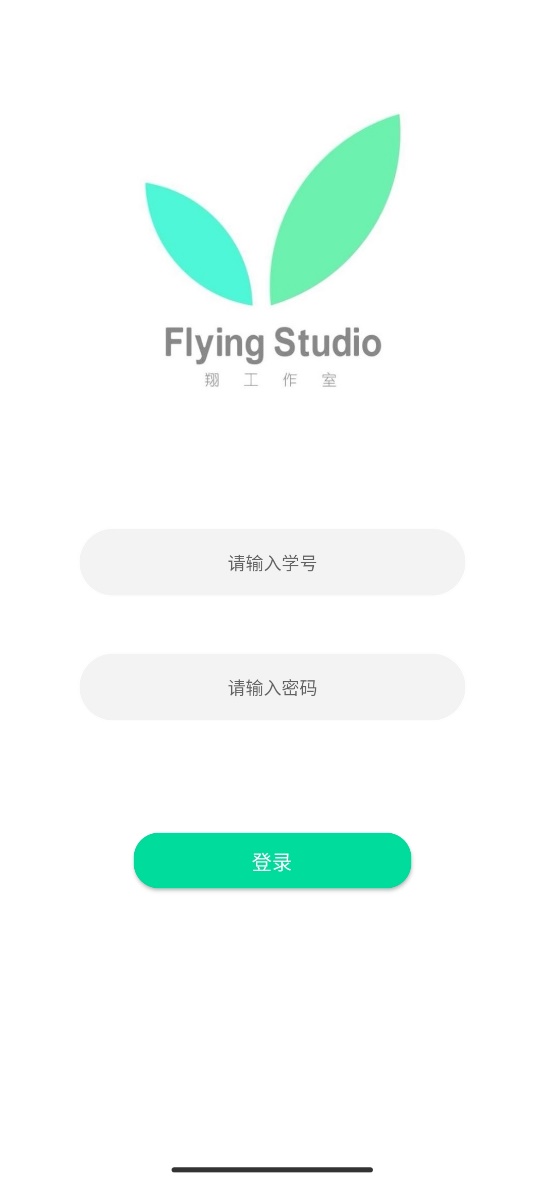
2 系统设计

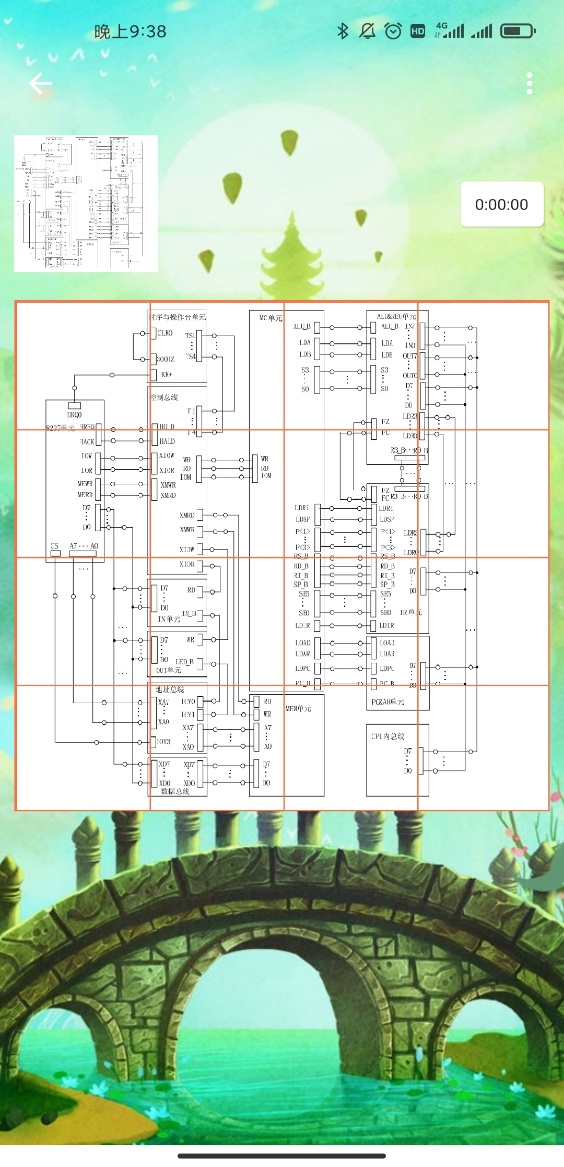
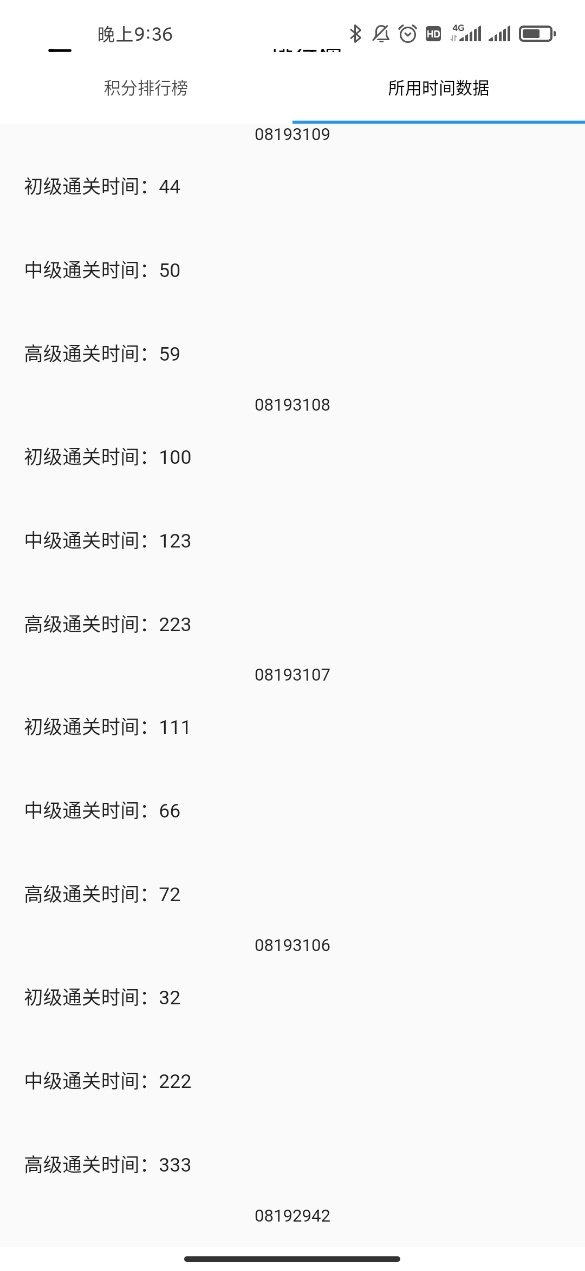
2.1 设计目标

本项目目标是设计开发一个拼图的游戏软件，软件能够自动对加载的图片进行分割，并打乱顺序后放置在不同的图片框中，用户使用鼠标点击图片框中的图片进行拼图，拼图成功后，系统会自动进行提示

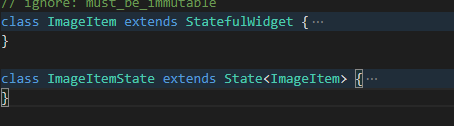
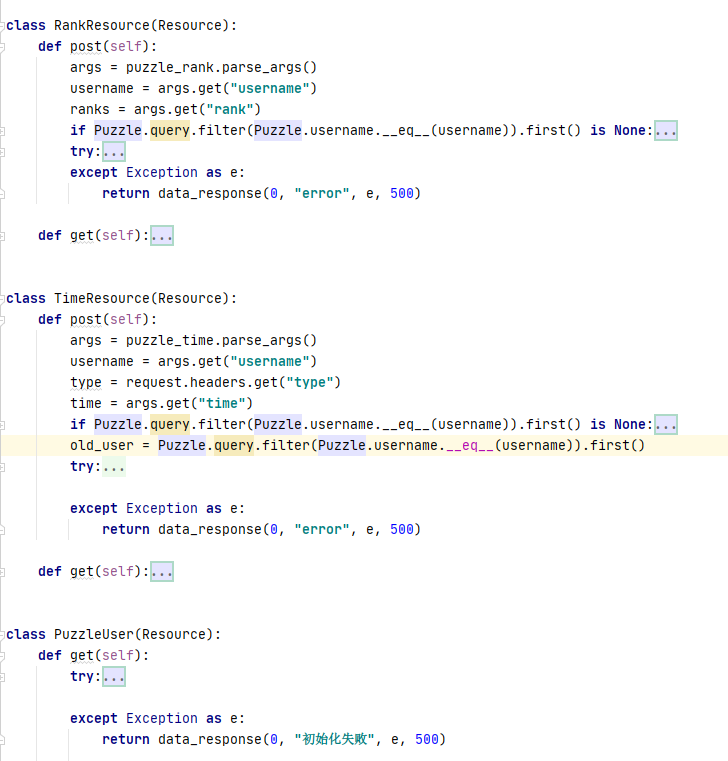
2.2 设计分析与算法流程



2.3 界面设计



2.4 关键类图



3 系统实现（运行调试）

代码：

Flask部分

puzzle\_init = reqparse.RequestParser()

puzzle\_init.add\_argument("username", type=str, help='请输入学号', required=True, location=['json'])

puzzle\_time = puzzle\_init.copy()

puzzle\_time.add\_argument("time", type=str, help='请输入是时间', required=True, location=['json'])

puzzle\_rank = puzzle\_init.copy()

puzzle\_rank.add\_argument("rank", type=str, help='请输入积分', required=True, location=['json'])

user\_fields = {

"username": fields.String,

"rank": fields.Integer

}

multi\_user\_fields = {

"num": fields.String,

"msg": fields.String,

"data": fields.List(fields.Nested(user\_fields))

}

time\_fields = {

"username": fields.String,

"time\_a": fields.Integer,

"time\_b": fields.Integer,

"time\_c": fields.Integer

}

multi\_time\_fields = {

"num": fields.String,

"msg": fields.String,

"data": fields.List(fields.Nested(time\_fields))

}

class RankResource(Resource):

def post(self):

args = puzzle\_rank.parse\_args()

username = args.get("username")

ranks = args.get("rank")

if Puzzle.query.filter(Puzzle.username.\_\_eq\_\_(username)).first() is None:

user = Puzzle()

user.username = username

db.session.add(user)

db.session.commit()

print("ok")

try:

old\_user = Puzzle.query.filter(Puzzle.username.\_\_eq\_\_(username)).first()

old\_user.rank = int(old\_user.rank) + int(ranks)

db.session.add(old\_user)

db.session.commit()

return data\_response(1, "success", old\_user.rank, 200)

except Exception as e:

return data\_response(0, "error", e, 500)

def get(self):

num = Puzzle.query.count()

user\_info = Puzzle.query.order\_by(Puzzle.rank.desc()).all()

msg = {

"num": num,

"msg": "获取成功",

"data": user\_info

}

return marshal(msg, multi\_user\_fields)

class TimeResource(Resource):

def post(self):

args = puzzle\_time.parse\_args()

username = args.get("username")

type = request.headers.get("type")

time = args.get("time")

if Puzzle.query.filter(Puzzle.username.\_\_eq\_\_(username)).first() is None:

user = Puzzle()

user.username = username

db.session.add(user)

db.session.commit()

old\_user = Puzzle.query.filter(Puzzle.username.\_\_eq\_\_(username)).first()

try:

if type == 'A':

if int(old\_user.time\_a) > int(time):

old\_user.time\_a = int(time)

if type == 'B':

if int(old\_user.time\_b) > int(time):

old\_user.time\_b = int(time)

if type == 'C':

if int(old\_user.time\_c) > int(time):

old\_user.time\_c = int(time)

db.session.add(old\_user)

db.session.commit()

return data\_response(1, "success", username, 201)

except Exception as e:

return data\_response(0, "error", e, 500)

def get(self):

num = Puzzle.query.count()

user\_info = Puzzle.query.all()

msg = {

"num": num,

"msg": "success",

"data": user\_info

}

return marshal(msg, multi\_time\_fields)

class PuzzleUser(Resource):

def get(self):

try:

args = puzzle\_init.add\_argument()

username = args.get("username")

if Puzzle.query.filter(Puzzle.username.\_\_eq\_\_(username)).first() is None:

user = Puzzle()

user.username = username

db.session.add(user)

db.session.commit()

else:

pass

return data\_response(1, "登录成功", username, 200)

except Exception as e:

return data\_response(0, "初始化失败", e, 500)

Flutter部分

// ignore: must\_be\_immutable

class ImageItem extends StatefulWidget {

ImageItem({

Key key,

this.index,

this.image,

this.rowRect,

this.targetRect,

this.offset,

this.color,

}): super(key: key);

final int index;

final ui.Image image;

final Rect rowRect;

final Rect targetRect;

Offset offset;

final Color color;

@override

ImageItemState createState() => new ImageItemState();

}

class ImageItemState extends State<ImageItem> {

Color color;

Offset offset;

Duration duration;

@override

void initState() {

super.initState();

color = widget.color;

offset = widget.offset;

duration = Duration();

}

@override

Widget build(BuildContext context) {

return AnimatedPositioned(

duration: duration,

curve: Curves.easeIn,

top: offset.dy,

left: offset.dx,

child: Container(

width: widget.targetRect.right - widget.targetRect.left,

height: widget.targetRect.bottom - widget.targetRect.top,

decoration: BoxDecoration(

border: Border.all(

color: color

)

),

child: Crop(widget.image, widget.rowRect, widget.targetRect),

),

);

}

updateColor(Color color){

this.color = color;

duration = Duration();

setState(() {

});

}

updateDuration(Duration duration){

this.duration = duration;

setState(() {

});

}

updateOffset(Offset offset){

this.offset = offset;

setState(() {

});

}

startAnimation(Duration duration, Offset offset){

this.offset = offset;

this.duration = duration;

setState(() {

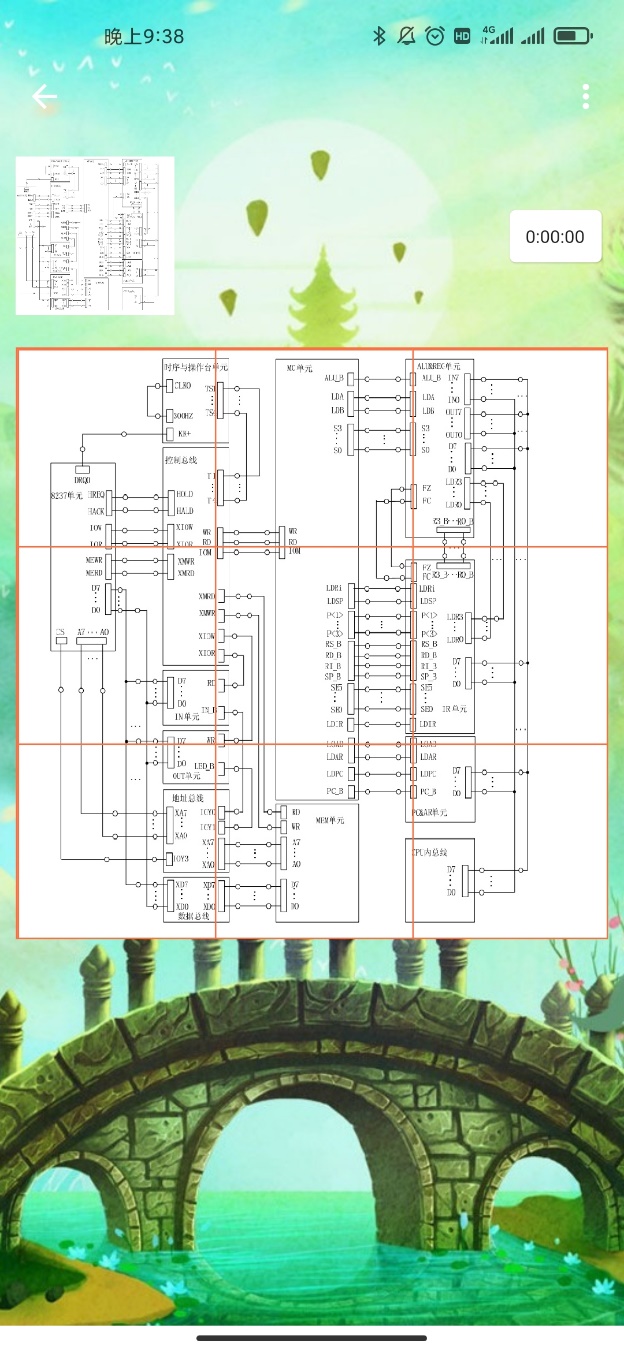
});

}

}

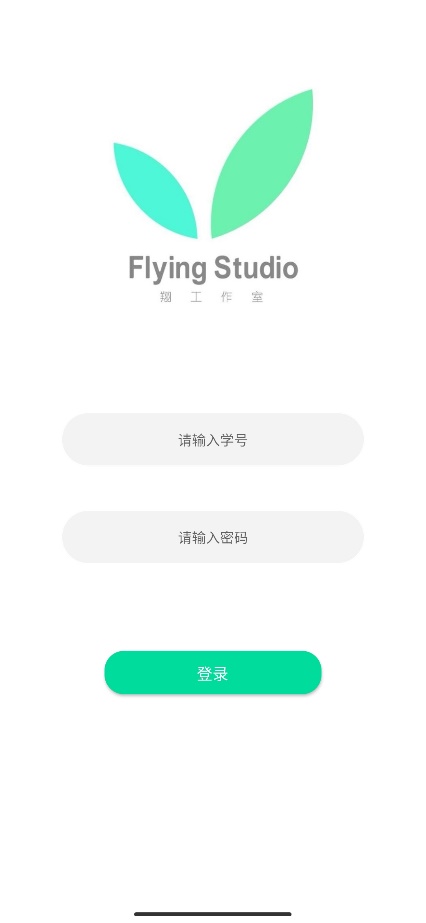
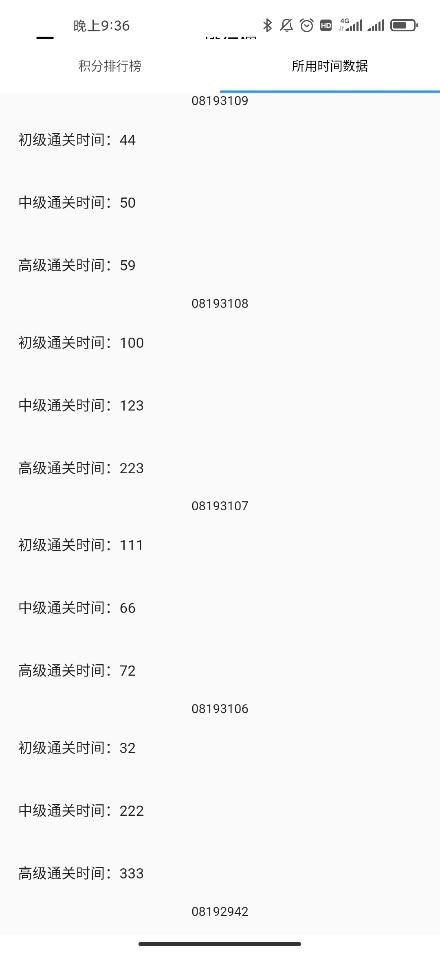
调试：

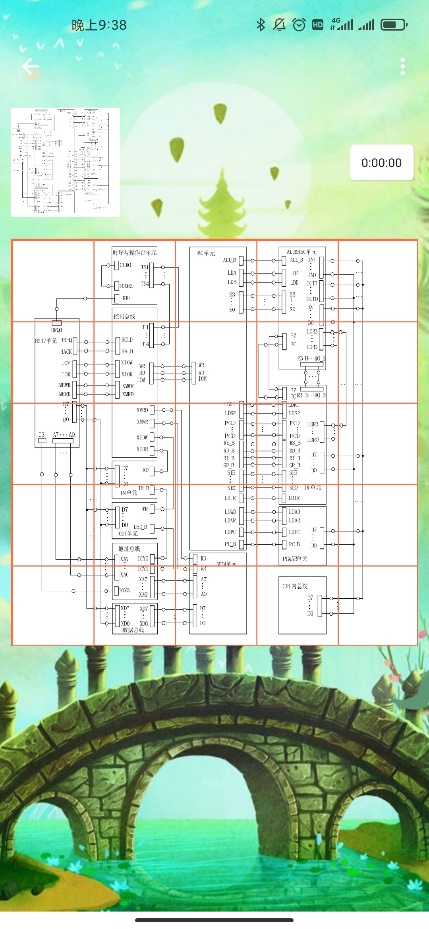
* 运行调试
* 图片尺寸自适应，动态分割，实时查看原图，图片自定义



* 

4 系统扩展

本项目实现了游戏难度分类，登录注册，游戏计时，游戏积分排行，用户最快游戏历史时长查询



5 总结

通过本次实验，我掌握了拼图游戏的设计方法，在编写代码的过程中，遇到了不少Bug，由于Flutter和Flask最近几年才出现，遇到的问题很多时候需要自己去试错。通过这次实验，我实现了一个简单的web应用，但由于各方面的原因程序还不是非常完善。以后有机会会开发更加健壮的程序/