详情页面

一.GBMViewDetailStoryboard

在Views/Storyboard中新建一个名为GBMViewDetail的storyboard。
大小设置为4英寸。
设置TopBar 为 Translucent Navigation Bar

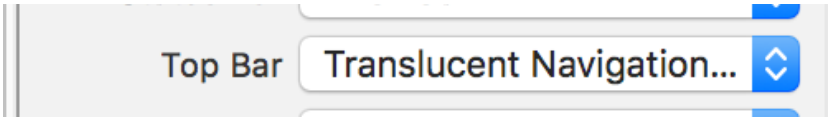

图 1-1 TopBar

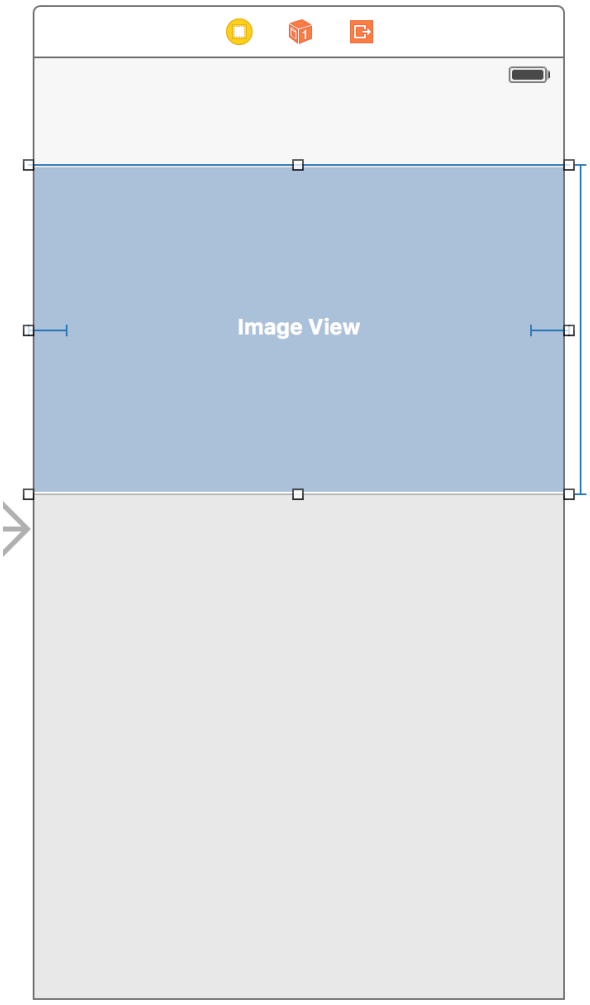

图 1-2 ImageView

在Scene中添加一个ImageView 放置到当中

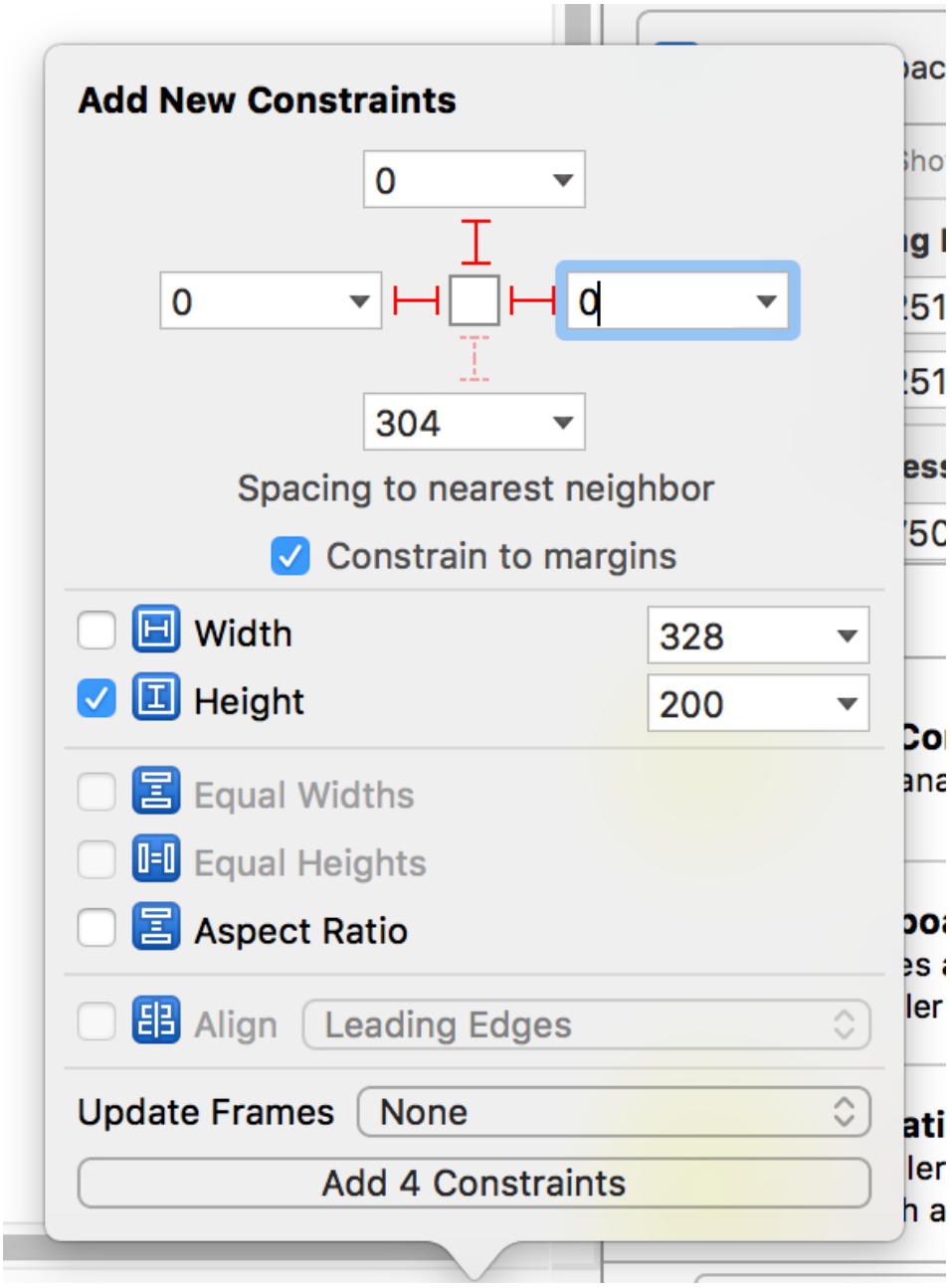

图 1-3 约束

并添加约束，向上距离为0，左右距离为0，高度为200。

到此，StoryBoard部分已经介绍完毕。

二.GBMViewDetailController

在Controllers/Square/ViewDetail中创建GBMViewDetailController。
将GBMViewDetail.storyboard与GBMViewDetailController进行关联。

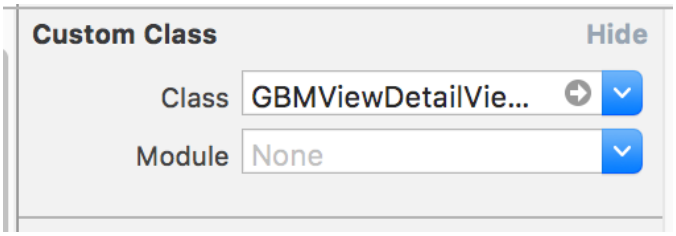

图 2-1 关联

在GBMViewDetailController中添加UITableViewDataSource,UITableViewDelegate协议

```
@interface GBMViewDetailViewController : UIViewController<UITableViewDataSource,
    UITableViewDelegate>
```

图 2-2 代理

讲SB中的UIImageView链接过来，并创建两个属性 pic_id指图片id，pic_url指图片网络地址。commentArr指评论的数组（当前版本只有一个评论）

```
13 @property (weak, nonatomic) IBOutlet UIImageView *PhotoImage;
14 @property (copy,nonatomic ) NSString *pic_id;
15 @property (copy,nonatomic) NSString *pic_url;
16 @property (nonatomic, strong) NSArray *commentArr;
17 @end
18
```

图 2-3 属性

在.m文件中定义成员变量。 UIActivityIndicatorView 与 UITableView。

```
@interface GBMViewDetailViewController ()
{
    UIActivityIndicatorView *activity;
     UITableView     *_tableView;

}
```

图 2-4 成员变量

在ViewDidLoad中添加一下代码，创建一个UIActivityIndicatorView，在图片从网上下载完成之前显示，下载完成之后停止。

```
activity = [[UIActivityIndicatorView alloc]initWithFrame:CGRectMake(0, 0, 30, 30)];

CGFloat width =self.view.frame.size.width/2;
[activity setCenter:CGPointMake(width , 160) ];
[activity setActivityIndicatorViewStyle:UIActivityIndicatorViewStyleWhiteLarge];
[self.view addSubview:activity];
[activity startAnimating];
self.PhotoImage.image = [UIImage imageWithData:[NSData dataWithContentsOfURL:[NSURL
    URLWithString:self.pic_url]]];
[activity stopAnimating];
```

图 2-5 UIActivityIndicatorView

然后将用户id，token以及图片id打包进入字典。

```
NSDictionary *dic = @{@"user_id":[GBMGlobal shareGloabl].user.userId, @"token":[GBMGlobal
    shareGloabl].user.token, @"pic_id":self.pic_id};
```

图 2-6 字典创建

```
NSDictionary *dic = @{@"user_id":[GBMGlobal shareGloabl].user.userId, @"token":[GBMGlobal
    shareGloabl].user.token, @"pic_id":self.pic_id};

GBMViewDetailRequest *request= [[GBMViewDetailRequest alloc]init];
[request sendViewDetailRequestWithParameter:dic delegate:self];
```

三.GBMViewDetailRequest

在Public/Network/Requests/Square/ViewDetail中创建GBMViewDetailRequest。

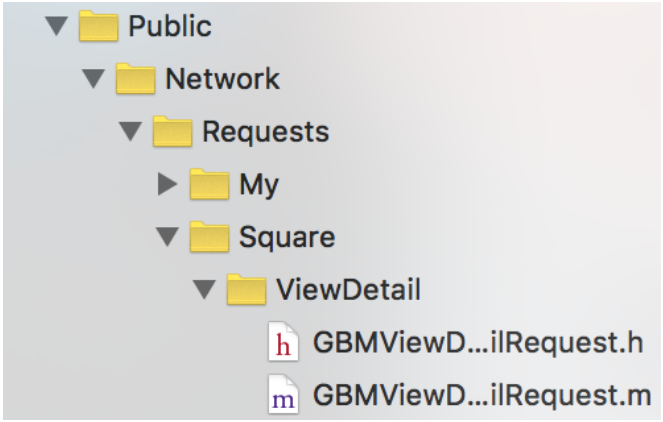

图 3-1 创建文件

与之前一样，在.h文件中创建delegate、属性以及方法。

```
@class GBMViewDetailRequest;

@protocol GBMViewDetailRequestDelegate <NSObject>

- (void)viewDetailRequestSuccess:(GBMViewDetailRequest *)request data:(NSArray *)array;

- (void)viewDetailRequestFailed:(GBMViewDetailRequest *)request error:(NSError *)error;

@end

@interface GBMViewDetailRequest : NSObject <NSURLConnectionDataDelegate>

@property (nonatomic, strong) NSURLConnection *urlConnection;
@property (nonatomic, strong) NSMutableData *receivedData;
@property (nonatomic, assign) id <GBMViewDetailRequestDelegate> delegate;

- (void)sendViewDetailRequestWithParameter:(NSDictionary *)paramDic
                                  delegate:(id <GBMViewDetailRequestDelegate>)delegate;
```

图 3-2 request的h文件

在.m文件中根据API文档

获取评论列表

/comment

GET获取。传user_id，token；可选参数pic_id（地点与图片是1:m的关系，图片与评论也是1:n的关系），若不传则返回所有节点的评论；可选参数page，页码，可选参数limit，每页条数，这两个参数用于控制分页，默认分别为1和10，即默认显示头10条。

图 3-3 API文档说明

可以得到链接的地址以及需要传递的参数。

```
NSString *urlString = [NSString stringWithFormat:@"http://moran.chinacloudapp.cn/moran/
    web/comment?user_id=%@&token=%@&pic_id=%@",paramDic[@"user_id"], paramDic[@"token"],
    paramDic[@"pic_id"]];
```

图 3-4 连接示例

在网络传输数据结束之后，创建GBMViewDetailParser类的parser对象并对接收的数据进行解析，放入数组中，最后调用代理中的viewDetailRequestSuccess:data:。

```
- (void)connectionDidFinishLoading:(NSURLConnection *)connection
{
    NSString *string = [[NSString alloc] initWithData:self.receivedData encoding:
        NSUTF8StringEncoding];
    NSLog(@"ViewDetail receive data string:%@", string);


        if ([_delegate respondsToSelector:@selector(viewDetailRequestSuccess:data:)]) {
        |       }
}
```

图 3-5 网络请求结束时

最后将这句添加到GBMViewDetailViewController中

```
NSString *token= [GBMGlobal shareGloabl].user.token;
 NSString *userId= [GBMGlobal shareGloabl].user.userId;
NSDictionary *dic = @{@"user_id":userId, @"token":token, @"pic_id":self.pic_id};

GBMViewDetailRequest *request= [[GBMViewDetailRequest alloc]init];
[request sendViewDetailRequestWithParameter:dic delegate:self];
```

图 3-6 完善GBMViewDetailViewController

到此为止网络请求Request就介绍完毕了。

四.GBMViewDetailParser

在Public/Network/Parsers中创建GBMViewDetailParser。

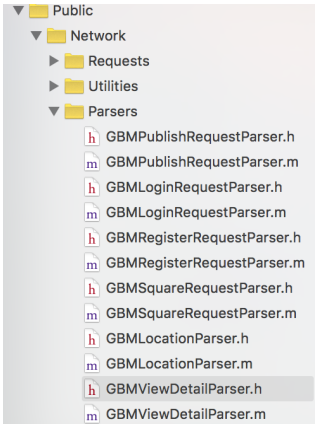

图 4-1 创建文件

在.h文件中创建属性与方法。

```
@property (nonatomic, strong) NSMutableArray *addrArray;
@property (nonatomic, strong) NSMutableArray *pictureArray;
- (NSArray*)parseJson:(NSData *)data;
```

图 4-2 h文件

在.m文件中进行解析，GBMViewDetailModel的创建可以查看第五节。

```
if ([[jsonDic class] isSubclassOfClass:[NSDictionary class]]) {
    id data = [jsonDic valueForKey:@"data"];
    for (id dic in data) {
        GBMViewDetailModel *model = [[GBMViewDetailModel alloc]init];
        [model setValuesForKeysWithDictionary:dic];
        [array addObject:model];
    }
}
```

图 4-3 解析

最后完善GBMViewDetailRequest请求。

```
- (void)connectionDidFinishLoading:(NSURLConnection *)connection
{
    NSString *string = [[NSString alloc] initWithData:self.receivedData encoding:
        NSUTF8StringEncoding];
    NSLog(@"ViewDetail receive data string:%@", string);
//

        GBMViewDetailParser *parser = [[GBMViewDetailParser alloc] init];
       NSArray *array = [parser parseJson:self.receivedData];
        if ([_delegate respondsToSelector:@selector(viewDetailRequestSuccess:data:)]) {
            [_delegate viewDetailRequestSuccess:self data:array];
        }
}
```

图 4-4 完善request

五.创建GBMViewDetailModel

在Models/Square/ViewDetail中创建GBMViewDetailModel

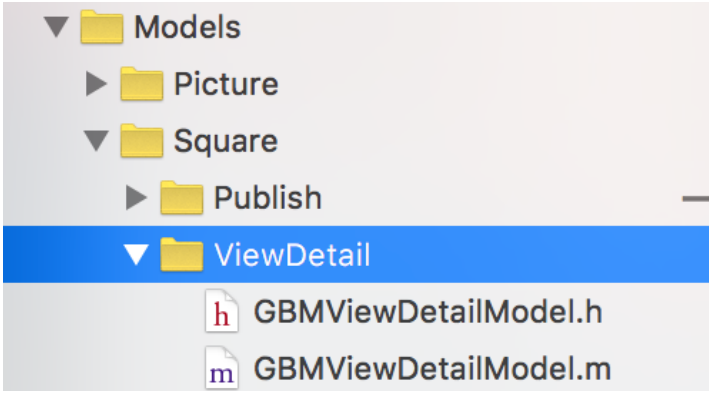

图 5-1 创建文件

在.h文件中创建属性评论（comment），时间（modified），setValueforUndefinedKey方法。

```
@property (nonatomic,strong) NSString *comment;
@property (nonatomic,strong) NSString *modified;
-(void)setValue:(id)value forUndefinedKey:(NSString *)key;
```

图 5-2 h文件

同时在.m文件中实现方法

```
@implementation GBMViewDetailModel

-(void)setValue:(id)value forUndefinedKey:(NSString *)key
{
}

@end
```

图 5-3 m文件实现方法

现在GBMViewDetailModel就创建完成了。

现在回到解析中，该解析的意思是每当解析到data中的一个数据，就创建一个GBMViewDetailModel,将得到数据的值赋值给GBMViewDetailModel，最后将model添加到数组中，再将数组进行返回。

```
if (error) {
    NSLog(@"The parser is not work.");
} else {
    if ([[jsonDic class] isSubclassOfClass:[NSDictionary class]]) {
        id data = [jsonDic valueForKey:@"data"];
        for (id dic in data) {
            GBMViewDetailModel *model = [[GBMViewDetailModel alloc]init];
            [model setValuesForKeysWithDictionary:dic];
            [array addObject:model];
        }
    }
}
return array;
```

图 5-4 解析步骤

到此为止解析器parser就介绍完毕了。

六.创建GBMCommentListCell

我们需要在查看图片的时候，同时查看评论，于是将图片用tableviewcell的方式创建。

创建GBMCommentListCell的时候我们不使用storyboard的方式，直接以代码的形式创建。

直接使用代码方式的话，很多16进制的颜色不能创建，所以引入了类别UIColor+AddColor，在设定颜色的时候就能调用方法，给Label等对象更加精确的颜色。

```objc
#import <UIKit/UIKit.h>

@interface UIColor (AddColor)

+ (UIColor *) colorFromHexCode:(NSString *)hexString;

@end
```

```objc
#import "UIColor+AddColor.h"

@implementation UIColor (AddColor)


+ (UIColor *) colorFromHexCode:(NSString *)hexString {
    NSString *cleanString = [hexString stringByReplacingOccurrencesOfString:@"#" withString:@""];
    if([cleanString length] == 3) {
        cleanString = [NSString stringWithFormat:@"%@%@%@%@%@%@",
                       [cleanString substringWithRange:NSMakeRange(0, 1)],[cleanString substringWithRange:NSMakeRange(0, 1)
                           ],
                       [cleanString substringWithRange:NSMakeRange(1, 1)],[cleanString substringWithRange:NSMakeRange(1, 1)
                           ],
                       [cleanString substringWithRange:NSMakeRange(2, 1)],[cleanString substringWithRange:NSMakeRange(2, 1)
                           ]];
    }
    if([cleanString length] == 6) {
        cleanString = [cleanString stringByAppendingString:@"ff"];
    }

    unsigned int baseValue;
    [[NSScanner scannerWithString:cleanString] scanHexInt:&baseValue];

    float red = ((baseValue >> 24) & 0xFF)/255.0f;
    float green = ((baseValue >> 16) & 0xFF)/255.0f;
    float blue = ((baseValue >> 8) & 0xFF)/255.0f;
    float alpha = ((baseValue >> 0) & 0xFF)/255.0f;

    return [UIColor colorWithRed:red green:green blue:blue alpha:alpha];
}

@end
```

图 6-0 UIColor+AddColor代码

（若不想麻烦可以省去这个步骤，添加颜色的时候使用系统自带的颜色即可）

首先在Views/Cell里面创建GBMCommentListCell
在.h文件中创建属性：

```
@interface GBMCommentListCell : UITableViewCell


@property(nonatomic,retain)UILabel     *usernameOfComment;
@property(nonatomic,retain)UILabel     *textOfComment;
@property(nonatomic,retain)UIImageView *headImageOfComment;
@property(nonatomic,retain)UILabel     *dateOfComment;
```

图 6-1 h文件属性

在.m文件中初始化cell，对usernameOfComment，textOfComment，headImageOfComment，dateOfComment进行设定。
首先创建成员变量

```
@interface GBMCommentListCell()
{
    CGFloat _cellWidth;
}

@end
```

图6-2 m文件中的成员变量

再创建initWithStyle:reuserIdentifier的方式创建cell，定义四个属性的位置颜色，头像的圆角，label的颜色以及大小。

```
-(id)initWithStyle:(UITableViewCellStyle)style reuseIdentifier:(NSString *)reuseIdentifier
{
    self = [super initWithStyle:style reuseIdentifier:reuseIdentifier];
    if (self) {
        _cellWidth = [UIScreen mainScreen].bounds.size.width;

        _headImageOfComment = [[UIImageView alloc] initWithFrame:CGRectMake(12, 10, 40, 40)];
        _headImageOfComment.backgroundColor = [UIColor blueColor];
        _headImageOfComment.layer.cornerRadius = 20;
        [self.contentView addSubview:_headImageOfComment];


        _usernameOfComment = [[UILabel alloc] initWithFrame:CGRectMake(12+60+9, 7, _cellWidth
            -24-60-18-100, 18)];
        _usernameOfComment.textColor = [UIColor colorFromHexCode:@"ee7f41"];
        _usernameOfComment.font = [UIFont fontWithName:@"Heiti SC" size:14];
        [self.contentView addSubview:_usernameOfComment];


        _textOfComment = [[UILabel alloc] initWithFrame:CGRectMake(12+60+9, 30, _cellWidth-24
            -60-18, 18)];
        _textOfComment.textColor = [UIColor colorFromHexCode:@"444444"];
        _textOfComment.font = [UIFont fontWithName:@"Heiti SC" size:14];
        [self.contentView addSubview:_textOfComment];


        _dateOfComment = [[UILabel alloc] initWithFrame:CGRectMake((_cellWidth-90-24), 7, 90,
            18)];
        _dateOfComment.textColor = [UIColor colorFromHexCode:@"9f9fa0"];
        _dateOfComment.font = [UIFont fontWithName:@"Heiti SC" size:14];
        [self.contentView addSubview:_dateOfComment];

    }
    return self;
}
```

图 6-3 cell中属性的设定

七.delegate在GBMViewDetailViewcontroller中的使用

根据前文，代理中有viewDetailRequestSuccess:data以及viewDetailRequestFailed:error:在网络请求完成后在GBMViewDetailcontroller中实现。
如果成功，则调用viewDetailRequestSuccess:data方法，commentArray等于传回来的数组，然后在ViewController中创建一个tableview，并设置代理，若失败，则调用viewDetailRequestFailed: error:，使UIActivityIndicatorView停止旋转。

```
-(void)viewDetailRequestSuccess:(GBMViewDetailRequest *)request data:(NSArray *)array
{
    self.commentArr = array;
    _tableView = [[UITableView alloc] initWithFrame:CGRectMake(0, self.PhotoImage.frame.size.
        height+self.navigationController.navigationBar.frame.size.height, self.view.frame.size
        .width, self.view.frame.size.height) style:UITableViewStyleGrouped];
    _tableView.delegate = self;
    _tableView.dataSource = self;
    [self.view addSubview:_tableView];


}
- (void)viewDetailRequestFailed:(GBMViewDetailRequest *)request error:(NSError *)error{
    [activity stopAnimating];
}
```

图 7-1 调用代理

使用tableview的代理，设定Section为1，行数为commentArr的数量，最后设定tableview的高度为60。

```
#pragma mark - UITableViewDelegate and UITableViewDataSource methods

-(NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
{
    return 1;
}

-(NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
{
    return self.commentArr.count;
}


-(CGFloat)tableView:(UITableView *)tableView heightForRowAtIndexPath:(NSIndexPath *)indexPath
{
    return 60;

}
```

图 7-2 设定TableView

最后创建cell，将commentArr中对应行数的评论和日期添加到cell中。

```
-(UITableViewCell *)tableView:(UITableView *)tableView
        cellForRowAtIndexPath:(NSIndexPath *)indexPath
{
    static NSString *CellIdentifier = @"cellIdentifier";
    GBMCommentListCell *cell = (GBMCommentListCell *)[tableView
        dequeueReusableCellWithIdentifier:CellIdentifier];
    if (cell == nil) {
        cell = [[GBMCommentListCell alloc] initWithStyle:UITableViewCellStyleDefault
            reuseIdentifier:CellIdentifier];
        cell.selectionStyle = UITableViewCellSelectionStyleBlue;
    }

    GBMViewDetailModel *model = self.commentArr[indexPath.row];
    cell.textOfComment.text = model.comment;
    cell.dateOfComment.text = model.modified;
    return cell;
}
```

图 7-3 创建cell

八.传递信息

在GBMsquareViewController中添加 toCheckPicture 方法，创建GBMViewDetailViewController对象detailVC，以sd_setImageWithURL的方式设定图片，然后给pic_id以及pic_url赋值，最后present出detailVC的视图。

```
- (void)toCheckPicture
{
    GBMViewDetailViewController *detailVC = VCFromSB(@"GBMViewDetail", @"detailVC");
    [detailVC.PhotoImage sd_setImageWithURL:[NSURL URLWithString:_pic_url]];
    detailVC.pic_id=_pic_id;
    detailVC.pic_url =_pic_url;
    [self.navigationController pushViewController:detailVC animated:YES];

}
```

图 8-1 赋值

最后在collectionView中的选择cell的代理方法中，添加[self.squareVC toCheckPicture] 调用该方法即可。

```
- (void)collectionView:(UICollectionView *)collectionView didSelectItemAtIndexPath:
    (NSIndexPath *)indexPath
{
    NSLog(@"%zd", indexPath.row);
    GBMPictureModel *pictureModel = self.dataArr[indexPath.row];
    self.squareVC.pic_url  = pictureModel.pic_link;
    self.squareVC.pic_id = pictureModel.pic_id;
    [self.squareVC toCheckPicture];
}
```

图 8-2 调用方法

详情页面就到此结束了。