Version <2.0>

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 17/11/2017 | 1.0 | Add details for Introduction, Architectural Goals and Constraints, Use-Case model and Logical View | Xuan-Vinh Nguyen,  Phu-Khoa Nguyen |
| 2/12/2017 | 2.0 | Add details for Deployment View & Implementation View | Phu-Khoa Nguyen |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

2. Architectural Goals and Constraints 4

3. Use-Case Model 4

3.1 Use-case: View food 3D model 4

3.2 Use-case: Share 4

3.3 Use-case: Log in with Facebook 5

3.4 Use-case: Log out. 5

3.5 Use-case: Scan menu. 5

3.6 Use-case: Read comment: 5

3.7 Use – case: Comment 5

3.8 Use-case: Rate 5

3.9 Use-case: Interact with model 5

3.10 Use-case: Customize model 5

3.11 Use-case: Order 5

3.12 Use-case: View ordered meals 5

3.13 Use-case: Handle orders 5

3.14 Use-case: View orders 5

3.15 Use-case: Finish order 5

3.16 Use-case: Serve orders 5

3.17 Use-case: Check paid 5

4. Logical View 6

4.1 Component: Authentication 6

4.2 Component: AR Camera 7

4.3 Component: Staff component 7

5. Deployment 7

6. Implementation View 7

# Introduction

This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system**.**

This application is being developed by REKT to mainly support displaying 3D model when scanning an appropriate picture. Besides, the program enables chefs as well as waiters to handle orders.

# Architectural Goals and Constraints

There are some key requirements and system constraints that have a significant bearing on the architecture. They are:

• The application provide tool for 3 different types of users: restaurant’s customers, chef and waiter. Therefore, there will be 3 different interfaces for this program.

• The list of orders on waiter and chef’s devices must be updated over time as customer orders meals. Hence, an implication of real-time database is indispensable.

• There would be no more than 1 second delay when scanning picture for displaying 3D model. Hence, all of the models will be stored offline.

• All performance requirements, addressed in the Vision Document [1], must be taken into consideration as the architecture is being developed.

# Use-Case Model



## Use-case: View food 3D model

This use-case describes how user views 3D model.

## Use-case: Share

This use case describes how user share the model on Facebook. It is an <<extend>> of view 3D model use-case.

## Use-case: Log in with Facebook

This use-case describes choice for user and admin to sign in application. It occurs when users/admins use the share function. This use-case is a ≪include≫ of use-case Share.

## Use-case: Log out.

This use-case describes how user logs out application. This use-case is a ≪extend≫ of use-case Share.

## Use-case: Scan menu.

This use case describes how the app scan menu. It occurs when users point their phone's camera into the menu. It is a <<include>> of use-case View food 3D model.

## Use-case: Read comment:

This use case describes how user read comment. It is an <<extend>> of view 3D model use-case.

## Use – case: Comment

This use case describes how user comment on a meal. It is an <<extend>> of read comment use-case.

## Use-case: Rate

This use-case describes how use rate a meal. It is an <<extend>> of view 3D model use-case.

## Use-case: Interact with model

This use-case describes how users interact with model. It is an <<extend>> of view 3D model use-case.

## Use-case: Customize model

This use-case describes how users customize model. It is an <<extend>> of view 3D model use-case.

## Use-case: Order

This use-case describes how users order meal. It is an <<extend>> of view 3D model use-case.

## Use-case: View ordered meals

This use-case describes how users view ordered meal. It occurs when users finish ordering. It is an <<extend>> of use-case Order.

## Use-case: Handle orders

This use-case describes how users handle ordered meal.

## Use-case: View orders

This use-case describes how users (restaurant staff) view ordered meals in order to handle them. It is an <<include>> of use-case Handle orders and Serve orders.

## Use-case: Finish order

This use-case describes how users view ordered meal. It is an <<include>> of use-case Handle orders.

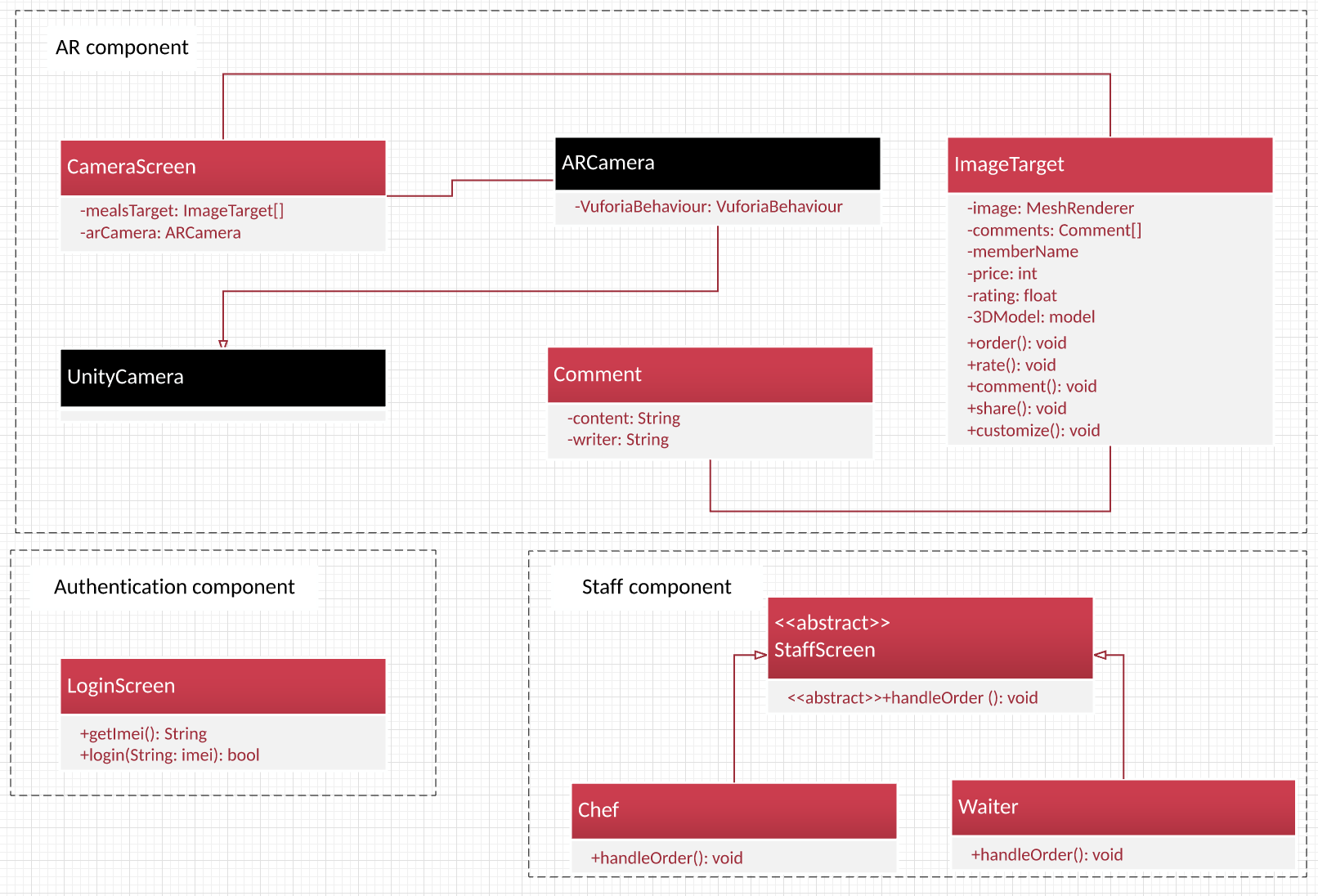
## Use-case: Serve orders

This use-case describes how waiter serves ordered meal.

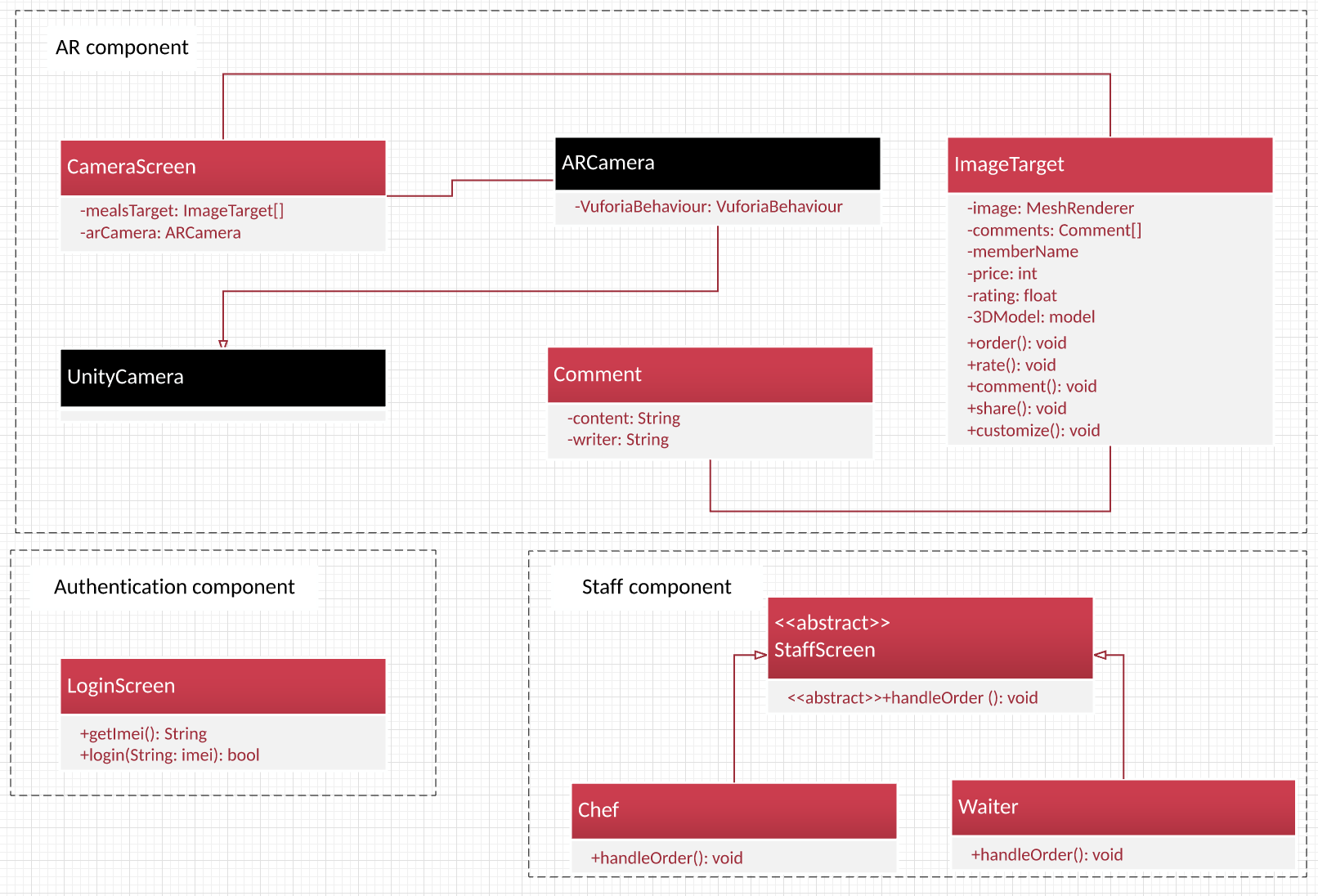
## Use-case: Check paid

This use-case describes how users view ordered meal. It is an <<extend>> of use-case Serve orders.

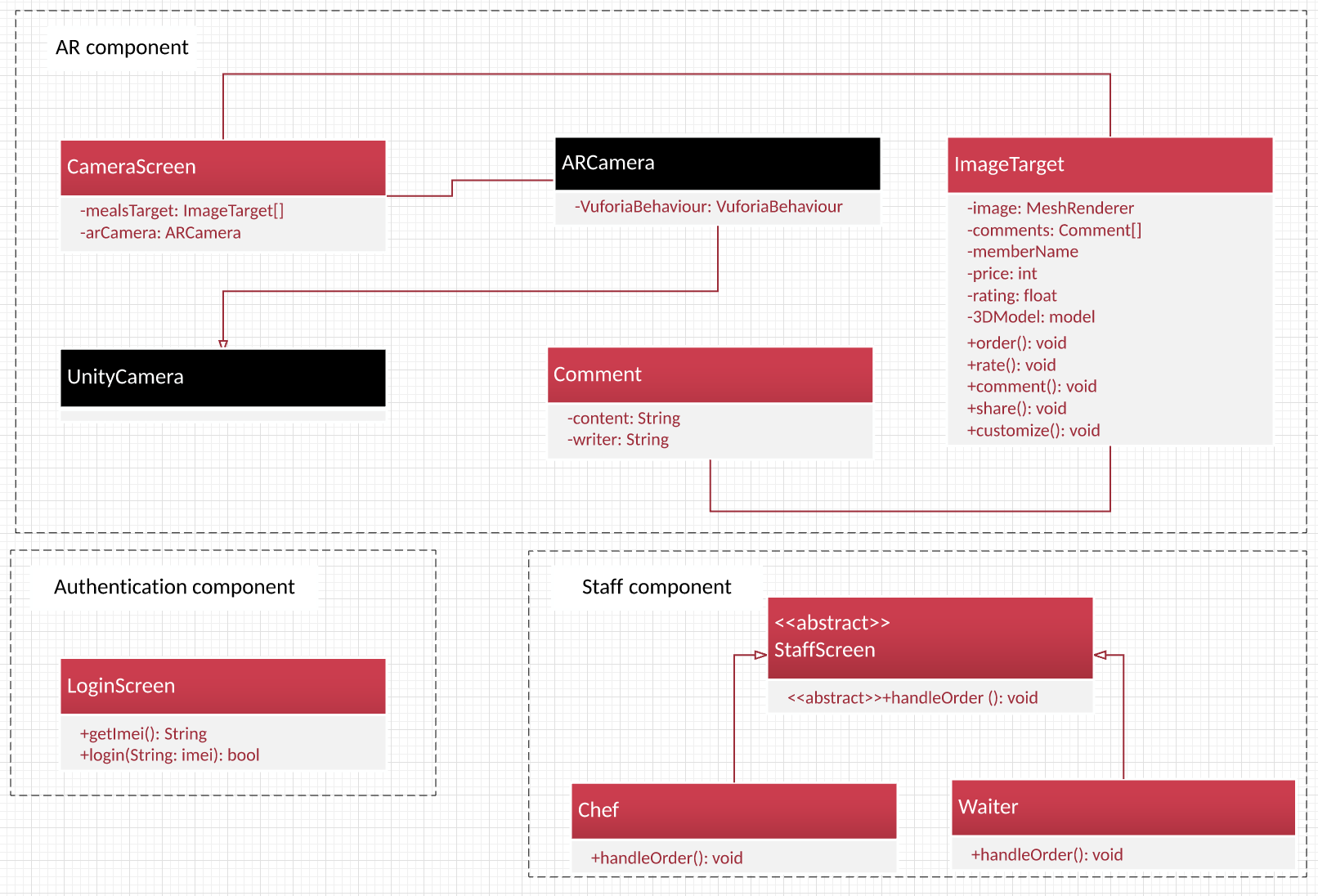
# Logical View



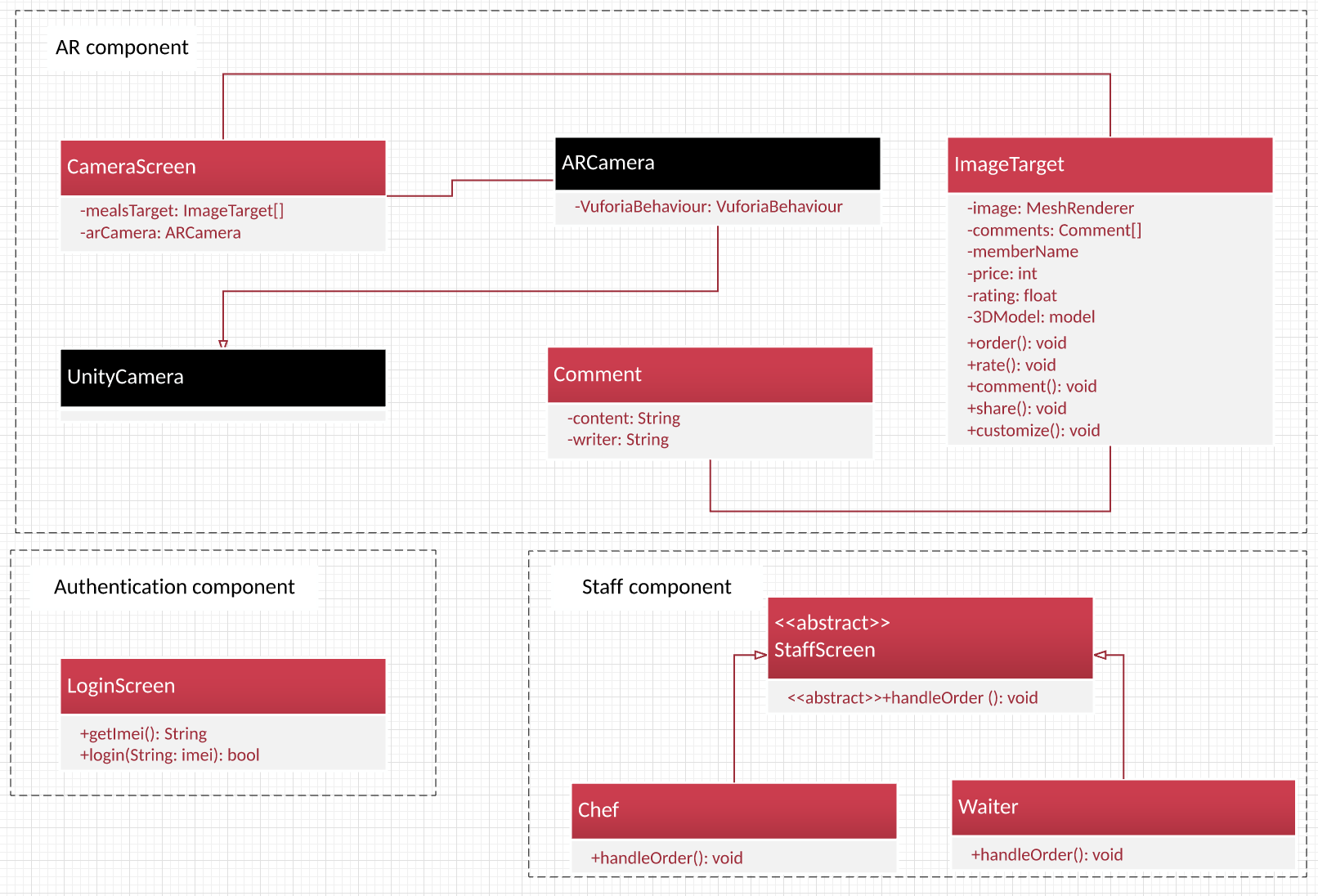
## Component: Authentication



## Component: AR Camera



## Component: Staff component

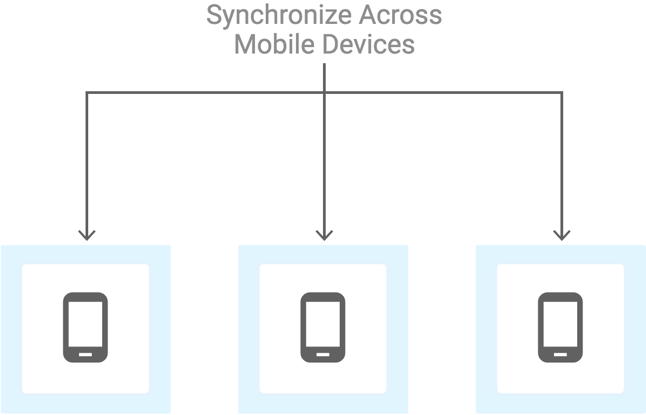


# Deployment



Push order information

Synchronization across devices



# Implementation View

* + - CS300 - Software Engineering
      * docs
        + design (contains design prototypes)
        + management (documents for management such as Weekly Report)
        + requirements (SAD, Use-case specs, Vision)
      * pa (containing all project assignments)
      * src
        + ARMenu (Unity project)

Assets

Scenes (application scene)

Scripts (implementation)

Library (Unity-generated)

obj (Unity-generated)

ProjectSettings (Unity-generated)

QCAR (Unity-generated)

UnityPackageManager (Unity-generated)

Other files (Unity-generated)