**Technical Document**

Deadline : 23 March 2018

Mission concept: “An augmented reality application to aid with home design”

**Basic design concept**

There will be two main different types of users:

* Homeowners/Public
* Decorators

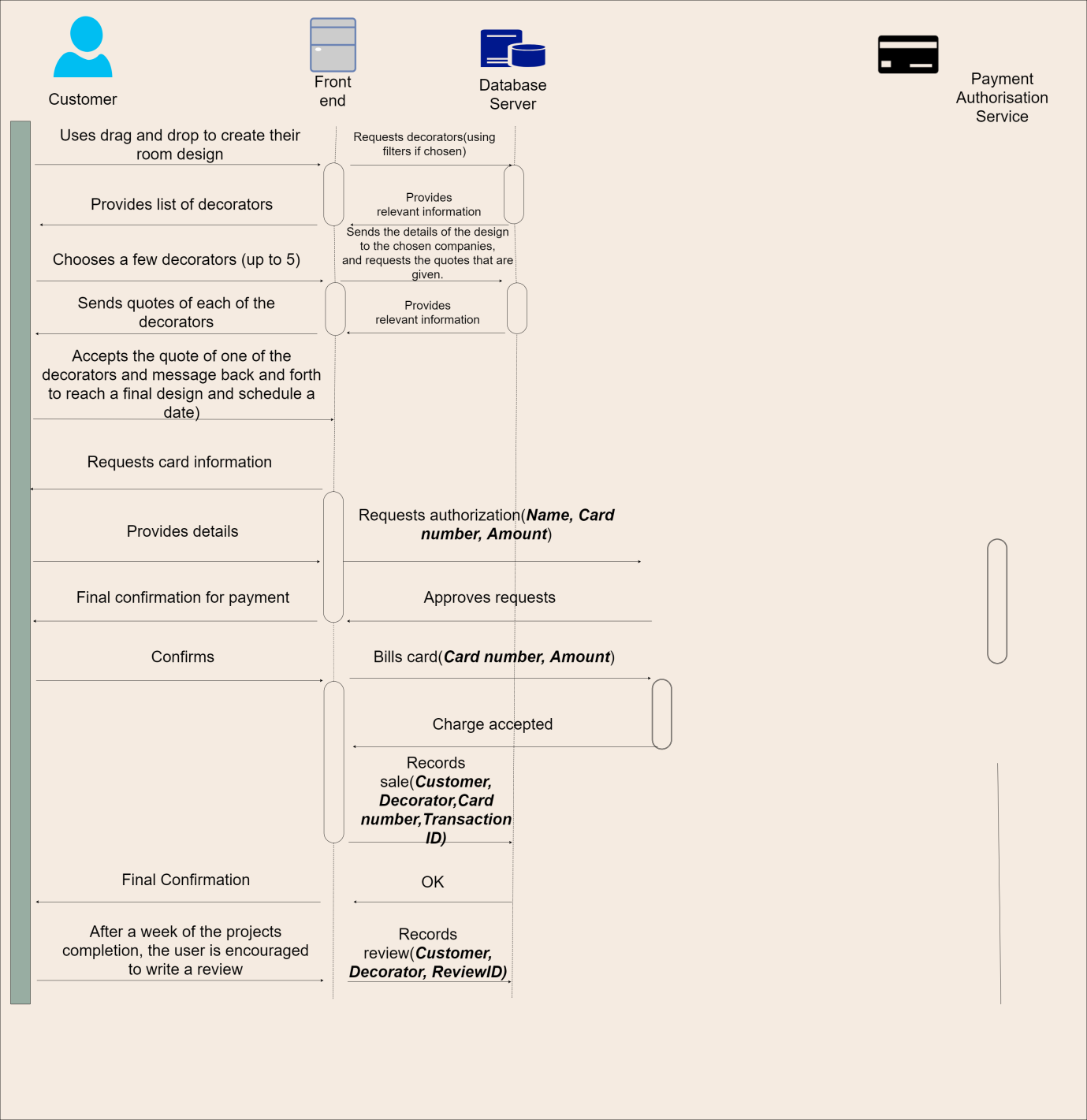
Using ‘Interiar’ homeowners will be able to visualize a space using Augmented Reality. Users have access to an IKEA catalogue of furniture etc. and can place different objects around a room in a style that suits them. The user can also change colours of walls and flooring. When the user is happy with a style they upload their work to the database. Decorators view the designs and contact the users via the In-app messaging system, if the decorator and user agree to a fee then the decorator does the work and the transaction is processed with a job fee for us.

**Level 1 – Ground work**

The application will be android based. This means that it will require a mix of coding languages:

* Unity C#
* Java
* Mongo

The database server is implemented using MongoDB and communicates with the Front-end of the system



**Payment System**



To process payments from the Users in our app a widely accessible and secure payment gateway is required. A Payment Gateway is an Application service provider for e-commerce that authorizes credit card or debit card payments for purchasing. A payment system is implemented in app using PayPal, and MasterCard API’s. The Android Mobile Payments Library SDK is implemented into the application. We use the PayPal Sandbox AppID. To accept we use Braintree Direct. We use a single payment process to execute one payment at a time. Through single payments you can receive immediate payment from a customer’s PayPal account or payment card. PayPal Express Checkout integration is most convenient for the application as it allows users to use their PayPal balance, bank account, or credit card to pay without sharing or entering any sensitive information on the application. PayPal Express checkout guides the user in the easiest way to process a payment. When the user clicks the PayPal button a PayPal popup window is opened, and the checkout process begins in the popup, the payment process is handled by PayPal but returns control to the app to finalize the payment, Where a confirmation page is shown.

**Augmented reality**

Our major feature of the app is the augmented reality drag and drop manoeuvrability. We intend to implement this using Unity3D. This is software that allows us to design a platform where the user can mix virtual elements into their physical surroundings. To begin with we began working with the Vuforia AR library which allowed us to use “markers” e.g. a book, that are used by the software to detect where the user wants to place objects. We found that to be quite a tedious task and not very convenient and we also noticed that as the range exceeded 3 metres, it quickly becomes incapable of consistently tracking the markers, and as such we have chosen to use the Wikitude library. This technology allows us to create markerless AR environments for our furniture, as through the use of their SLAM (Simultaneous Localisation and Mapping) algorithms we can recognise features of a room,as well as their relative position and, based on the movement of the user, align the augmented content accordingly and so the user can seamlessly drop their furniture anywhere around the room that they want instantly. To begin with the library allows us to create simple objects such as these shown.

The user has the option to also turn their camera portrait or landscape depending on the need. As well as this ,should the user decide to turn the camera around to the rest of the room after placing furniture into their desired space,the objects will remain linked to that space and once the user decides to look back to that area of the room , the furniture will pop back up in their spots. The assets (furniture) that are provided are simplistic models from wikitude, however the models that we intend to implement are from the ikea data set, which would look more true-to-life.

Will be updated as task moves along