

**Software design specification**

**for Putiton team’s SRH Game**

Version: 0.0.1 (Draft)

Date: 27 Feb. 2016

Author: SENSMEDIA SRL

**Version History**

|  |  |  |  |
| --- | --- | --- | --- |
| Ver. | Date | Summary of Changes | Author |
| 0.0.1 | 27-Feb-16 | Creation of initial draft version | SENSMEDIA SRL |
|  |  |  |  |
|  |  |  |  |

Contents

[1 Introduction 4](#_Toc444450330)

[1.1 Purpose 4](#_Toc444450331)

[1.2 Scope 4](#_Toc444450332)

[1.3 Glossary 4](#_Toc444450333)

[2 Deployment Diagram 5](#_Toc444450334)

[3 Architectural Design 5](#_Toc444450335)

[3.1 Description 5](#_Toc444450336)

[3.2 Domain model 6](#_Toc444450337)

[4 Use Cases 7](#_Toc444450338)

[4.1 Mobile quiz game 7](#_Toc444450339)

[4.2 Quiz/Content manager module 8](#_Toc444450340)

[4.3 Analytics/Statistics module 8](#_Toc444450341)

[5 User interface 9](#_Toc444450342)

[6 Technology stack 9](#_Toc444450343)

[6.1 Technology used 9](#_Toc444450344)

[6.2 Security 9](#_Toc444450345)

# ****Introduction****

## Purpose

This document contains high level architecture, use cases and technology stack of Putiton team’s project. Describes the implementation of an application that creates, manages and delivers quiz games.

## Scope

The contents of this document were chosen following a pragmatic approach. Miscellaneous details which previously didn’t brought real value are excluded. E.g. sequence diagrams where better described in direct communication with the client during periodic demos/milestones presentations.

The product will consist of three logical modules:

1. The mobile game
2. The data aggregator/statistics/analytics module
3. The web application for creating and managing quizzes will use and present data from the analytics module

The Module 1 will be definitely physically separated from the other two. During implementation it will be decided if 2 and 3 will be separated as well, for independent evolution.

## Glossary

Web Services – "Web services" describes a standardized way of integrating Web-based applications

Rest web services – RESTful Web Services are REST architecture based web services. In REST Architecture everything is a resource. RESTful web services are light weight, highly scalable and maintainable and are very commonly used to create APIs for web based applications.

Hybrid mobile app – mobile applications created using web technologies, that run on multiple platforms (e.g. iPhone, Android)

Server – computer program that manages access to a centralized resource or service in a network

Database – a structured set of data held in a computer that is accessible in various ways

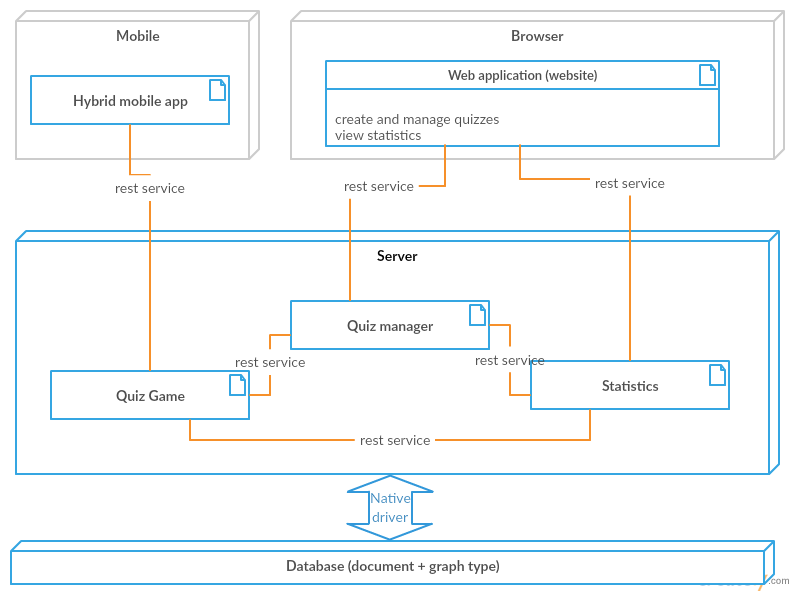
Graph database – database that uses graph structures for semantic queries with nodes, edges and properties to represent and store data

Document database (NOSQL) – database which uses document data model, where each record and its associated data is thought of as a “document”

Horizontal scaling – To scale horizontally (or scale out/in) means to add more nodes to (or remove nodes from) a system, such as adding a new computer to a distributed software application

Stateless application – A stateless app is an application program that does not record data generated in one session – such as information about user settings and events that occurred -- for use in the next session with that user.

# Deployment Diagram



Hybrid mobile app will be packed for iPhone and Android. Other Smartphones will be limited to using mobile website equivalent.

Database and Server can be easily scaled horizontally. Application are stateless.

# Architectural Design

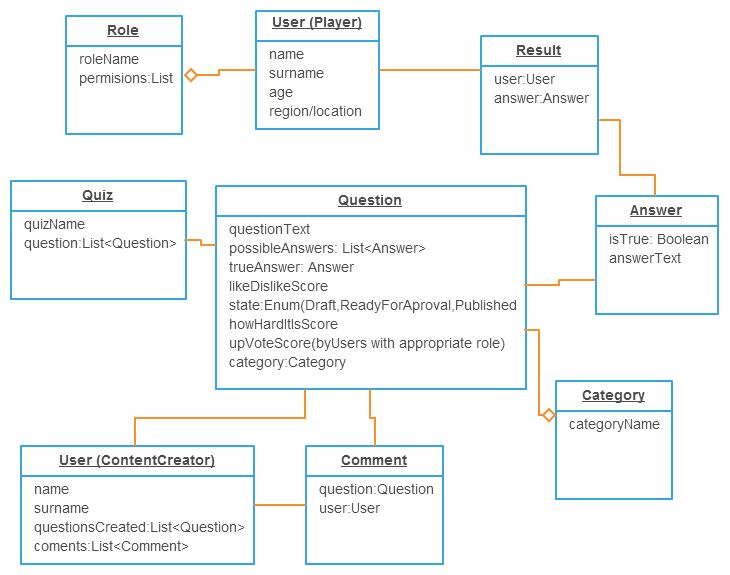
## Description

The architecture of the application will respect the following rules:

* Easy to understand and close to real life model
* Logical modules can evolve independently (development, building and deployment of one module can be done without bothering about other modules)
* Modules are designed with maximal reusability in mind (are constructed from the start in a service-oriented manner)
* The application should easily scale up vertically and horizontally
* Cost effectiveness of implementation and maintenance
* Use of open standards and open source software

## Domain model

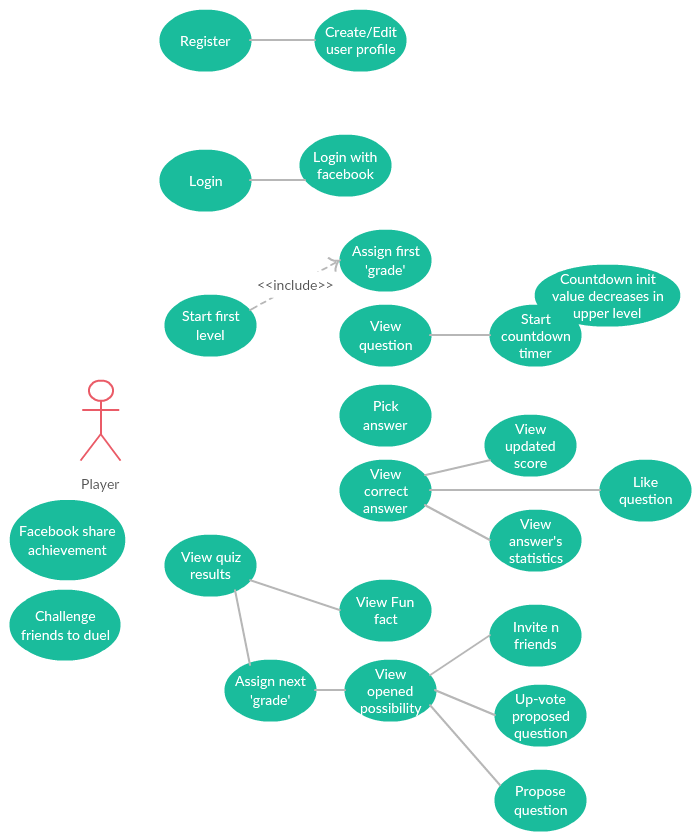
The following diagram describes the domain model which will be reflected in the database and in the application layer as well.



Objects/Classes and their relationships (Fig 1)

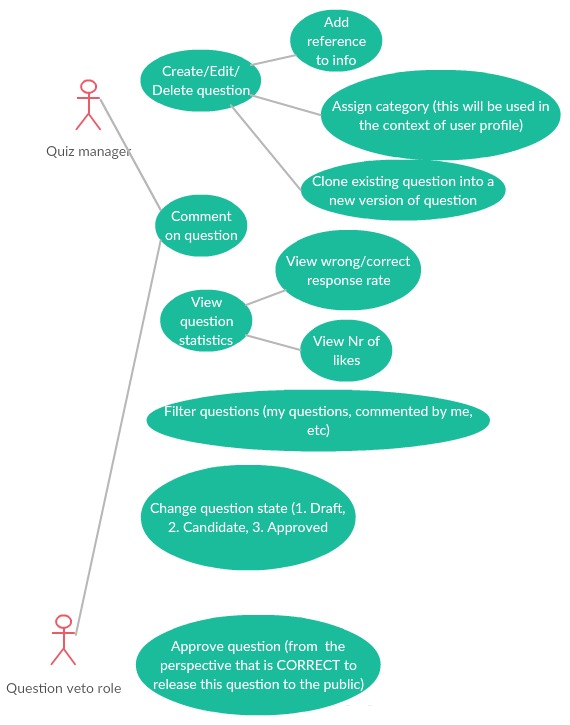
# Use Cases

## Mobile quiz game



Mobile game functionality (Fig. 2)

## Quiz/Content manager module



Use cases of content management module (Fig 3)

## Analytics/Statistics module

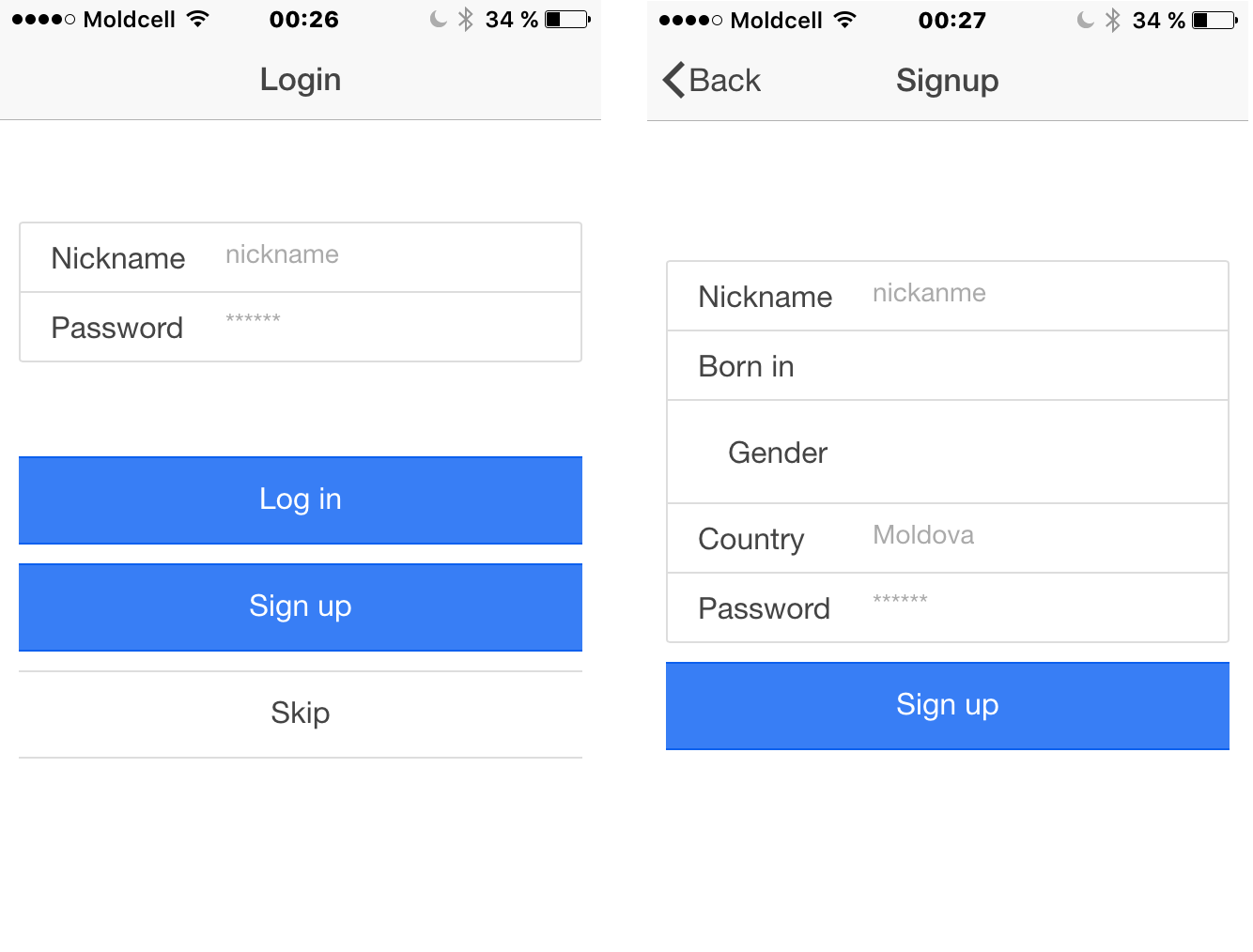
Answers Statistics – per user metadata (e.g. age)

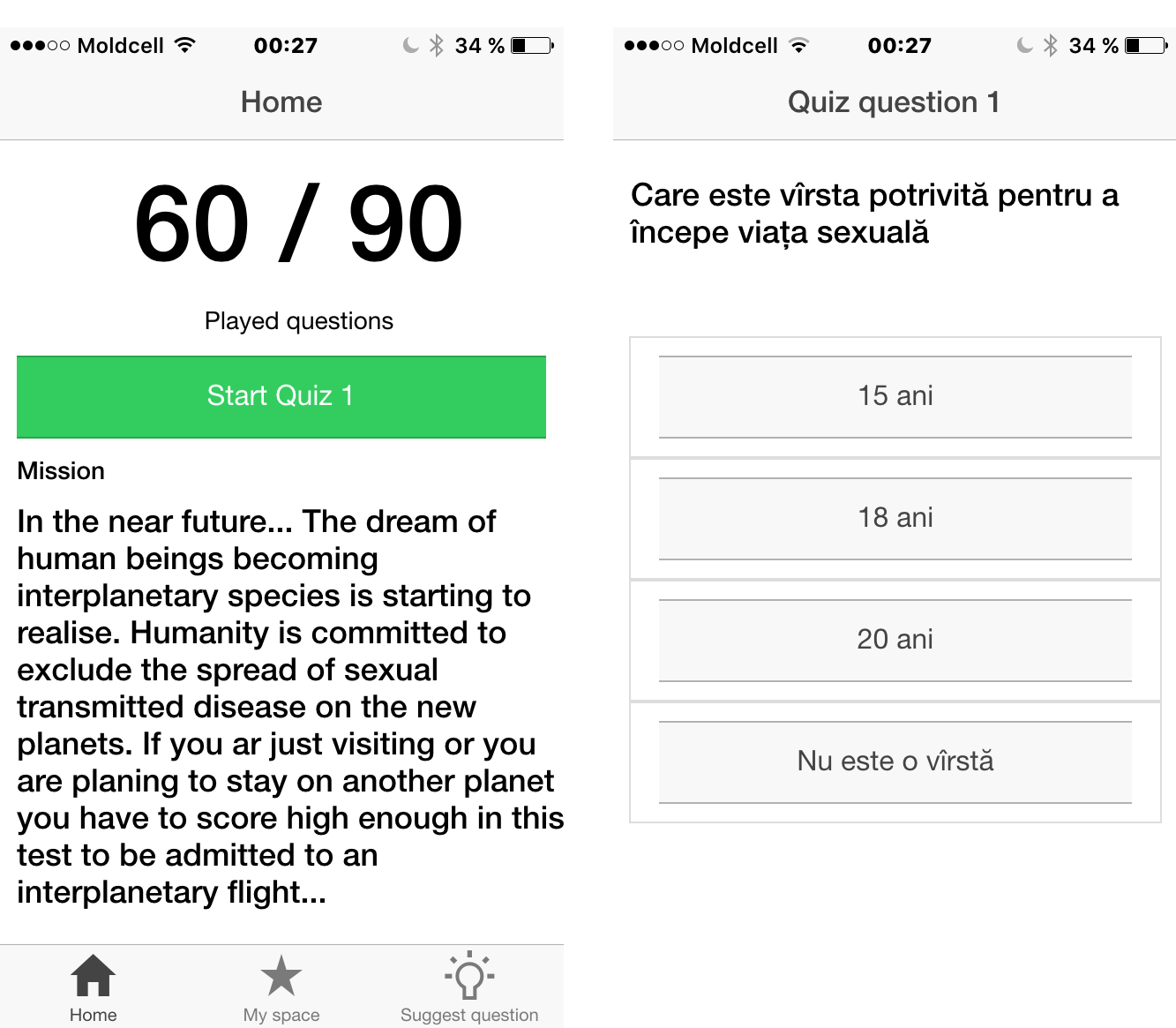
Question statistics – ‘how liked’ is it

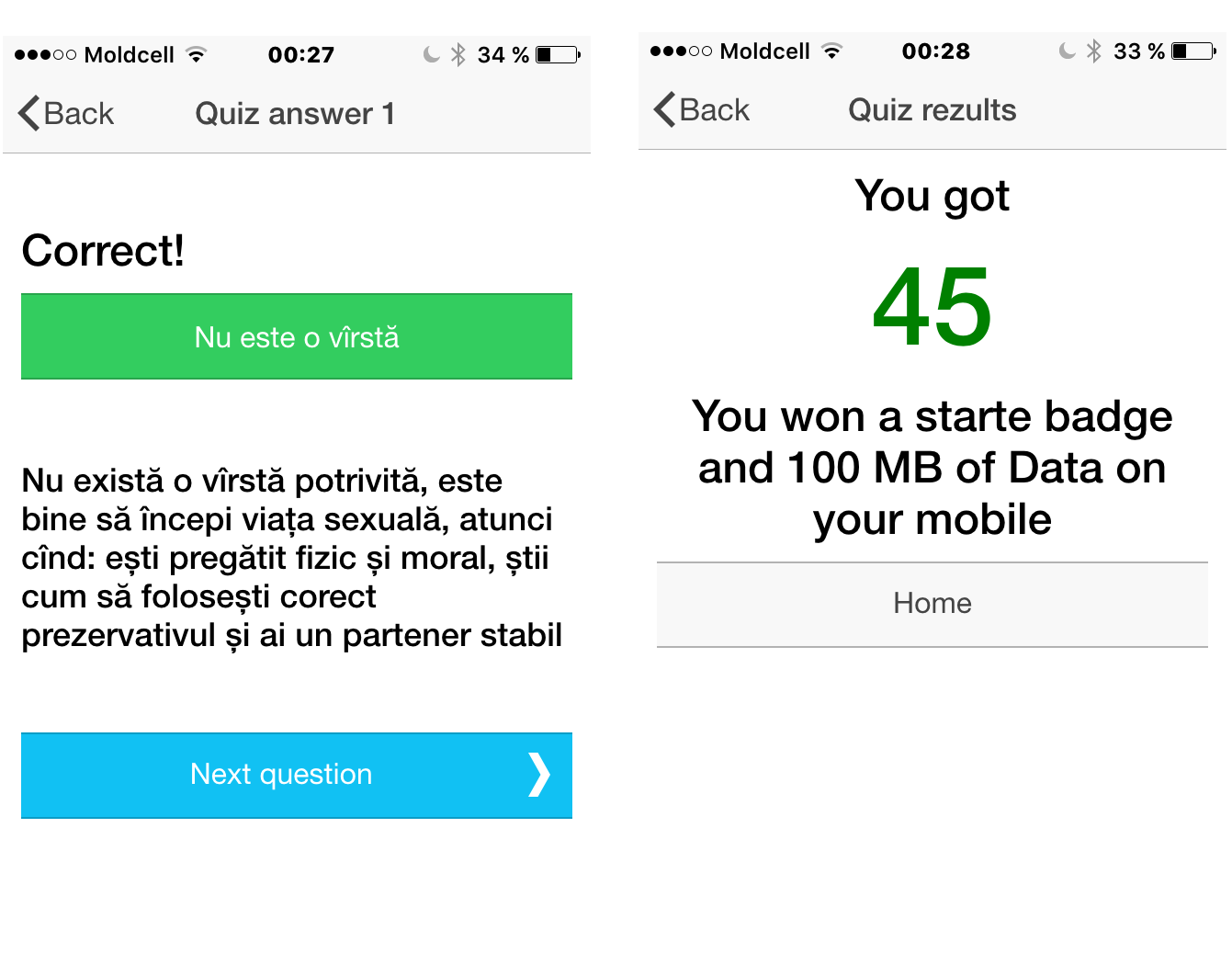
Use base statistics – current active users, new users per period of time, users progress etc.

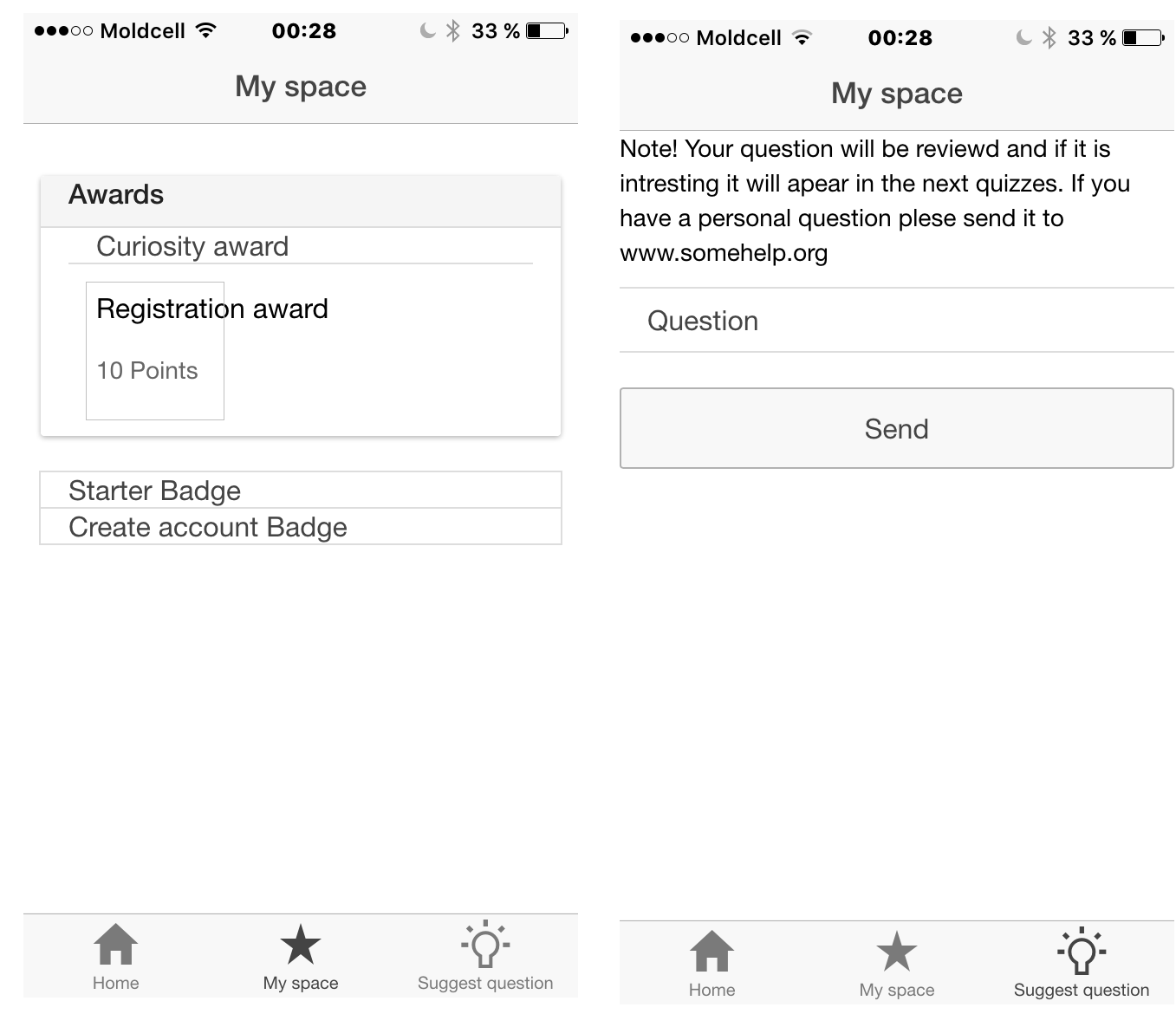
# User interface

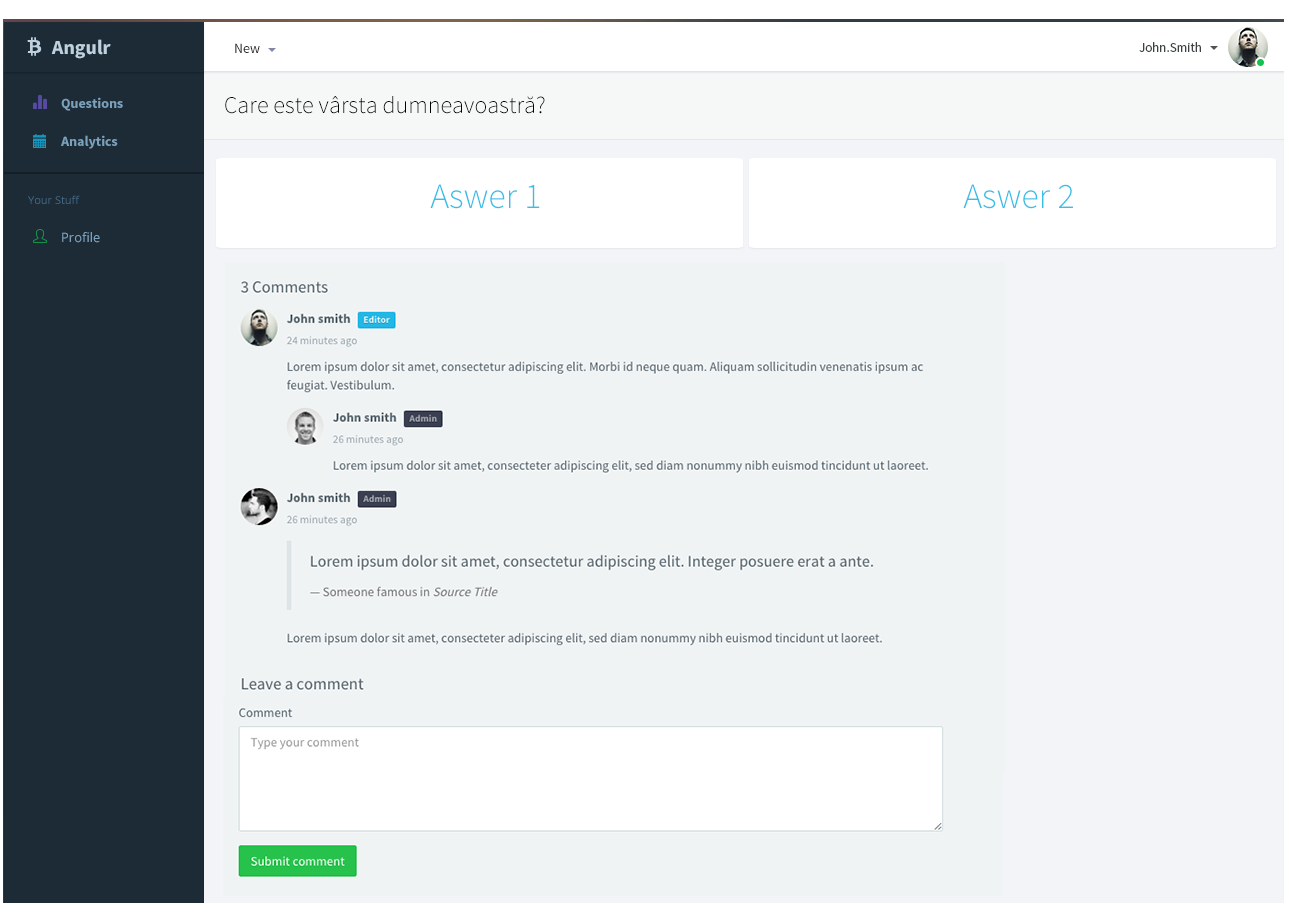
See below the list of screens (a part of them are mocked a part of them are taken from the second demo mobile app):

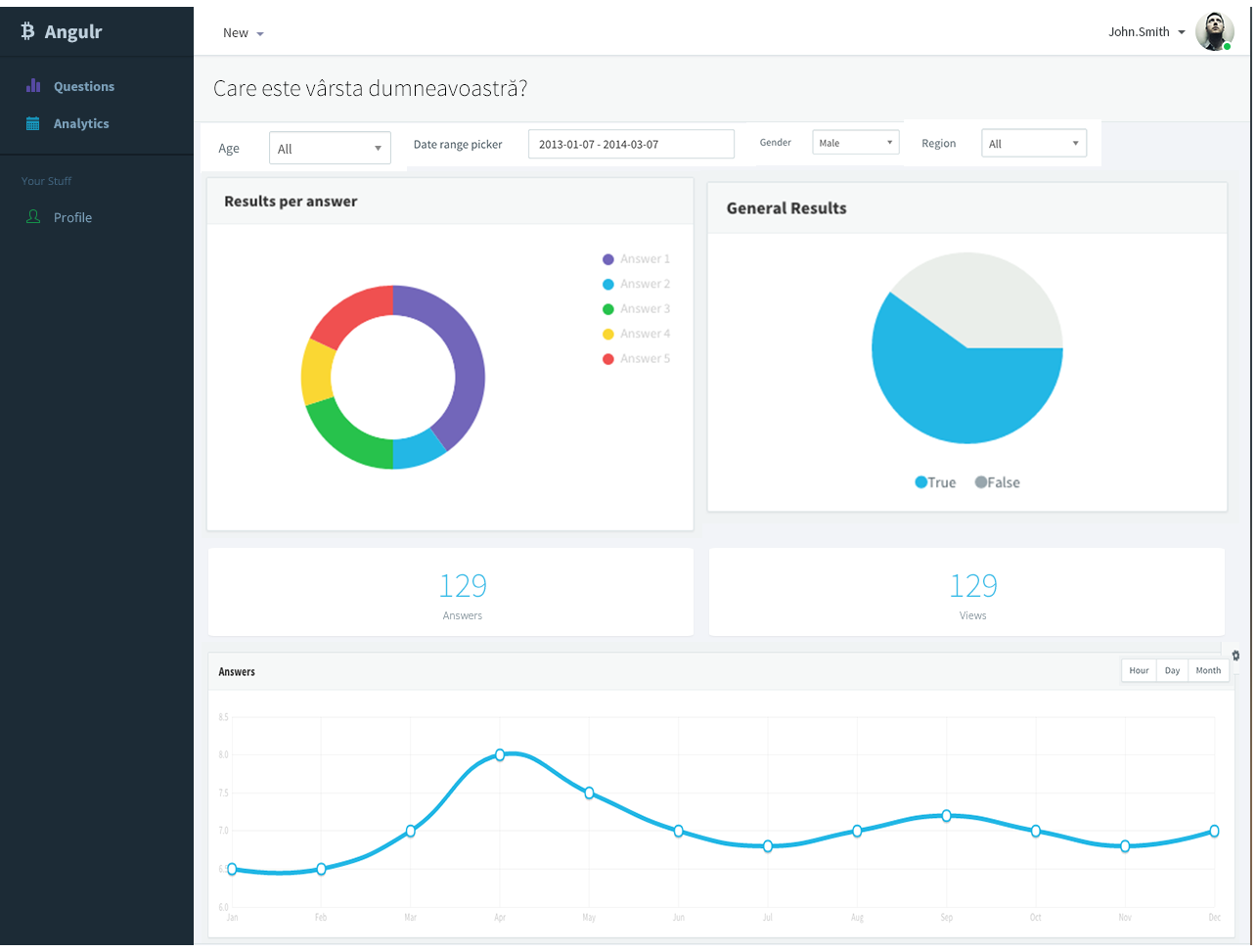












# Technology stack

## Technology used

Nodejs – easy scalable, fast and efficient for concurrent use; easy building and testing web services; shares the development language with the front end

<https://nodejs.org/en/>

Ionic Framework – build cost effective hybrid mobile applications, uses well known web development technologies

<http://ionicframework.com/>

AngulaJS – front end web applications (websites) development framework

<https://angularjs.org/>

OrientDB – document based with graph database capabilities; has a convenient ‘SQL like’ query syntax

## Security

JWT.IO – library for using JSON Web Token for authentication and authorization in a stateless way

<https://jwt.io/>