**Team: Space Wolves**

**Universum.kz**

**Software Test Document (STD)**

**Kabduali Dulat**

# Introduction

This document is the test plan document for the universum.kz web application. It will outline all testing strategies to be used. The plan will outline what features need to be tested, the types of tests that will be performed, and the team responsible for each phase of testing. The document will also include the risks associated with the plan.

**1.1 Objectives**

The test suite should provide test coverage(in our case codecov.com service ) metrics and verify that each section is working as intended. The test should verify that a section is ready to be deployed in the field as soon as the test is completed.

**1.2 Testing Strategy**

The following will be included in the test plan as a standard template for how each

section must be tested to meet the appropriate standards set by this document.

*Specific test plan components include:*

* *Purpose for this level of test,*
* *Features to be tested,*
* *Features not to be tested,*
* *Management and technical approach,*
* *Pass / Fail criteria,.*

**1.3 Scope**

Testing will be performed as a continuous integration process throughout the life cycle as the product is in production. Due to the involved nature of testing, test planning will be a continuous process that changes in scope alongside the products development. This test plan template will be updated alongside the overall product and will reflect any changes in scope, design, and time schedule. Test plans must be developed for each level of product testing.

**2. Test plan**

# 2.1. Features To Be Tested

The following features are the major functional capabilities of the project that need to be tested at all phases of the testing cycle.

1. Rating the university
2. Writing the review
3. Calculation of overall university rank score
4. Showing the universities by rank order and other filters
5. Register new user
6. Login
7. s

# 2.2. Features Not To Be Tested

# 2.3 **Testing tools and approach**

(This section needs an overall system diagram that will be outlined in the final SSD)

The testing for the universum.kz system will be primarily unit testing designed to match each of the requirements. The software developers will be in charge of the unit testing and must submit a test scripts. The testing servers (travis-ci.org/com in our case) will automatically test the new feature realeases.

System continous integration will be performed by the entire team and the developers are in charge of insuring that their parts work with each other. System integration will not be allowed to begin until the developers have completed their unit testing. A component cannot have any major defects when system integration begins for that specific component.

# 2.3.1 Testing Materials

In our project development we fully use CONTINOUS INTEGRATION principle , so we use the thir-party solutions like Travis-ci to automatically test and build project when each team member commits new version of the features or whole project on the github.com.

**2.3.1.1 Hardware**

The CI systems need the testing server . We use the third-party solution: travis-ci.org and codecov.com to test automatically test and built the project realeases on their servers.

**2.3.1.2 Software**

As project's automatically tested and built on third-party servers by the their software solutions , we don't need any software requirements by our side, only the test scripts to tell the server what to test and which requirements the server will to build the new project realese on their servers.

**3. TEST CASES**

**Case – N :**

# 3.N Pass / Fail Criteria

The following section outlines what constitutes a successful or failed test. This will include when a section is in a state of critical error that the remaining scheduled testing must be suspended and under what conditions it may resume.

**3.N.1 Suspension Criteria**

If a test in the basic function of each of the features to be tested failed then all testing on that field must be suspended until they are fixed. For example, if the basic add medicine function fails then all systems related to that feature such as alarms for medicine must be suspended until that feature is fixed.

**3.N.2 Resumption Criteria**

The testing may resume on a section once a developer indicates that they have fixed a specific error. This indication must include both new code and a detailed description of how the bug was fixed.

**3.N.3 Approval Criteria**

For a test to be considered a pass it must meet the following criteria

1. Successfully demonstrates the designated function is working
2. Demonstrates that edge cases such as reaching the maximum number of characters in a medicine name is handled
3. Returns successful error responses to the designated features.

**Case - 1:**

Purpose of this feature is to register new user by pressing the

entering new user information and pressing submit button.

**Inputs: 1)** User major registration information.

**2)** User additional information.

**Expected outputs and PASS/FAIL criteria:**

**1) FAIL** IF user can not submit (press the submit button) on the client side. Otherwise: **PASS**

**2) FAIL** If user **did** submitted but server did not save the infor to the model. Otherwise : **PASS**

**Case -2:**

Purpose of this feature is to rate the university or the course by pressing the

rating button on the university/course page.

**Inputs: 1)** User major registration information.

**2)** User additional information.

**Expected outputs and PASS/FAIL criteria:**

**1) FAIL** IF user can not login (press the login button) on the client side. Otherwise: **PASS**

**2) FAIL** If Django\_Rest server doesn't respond to angular client while logining Otherwise : **PASS**