Faculty of Informatics
Masaryk University

PA181

Services - Systems, Modeling and Execution

Term Project Documentation

Adrián Tóth (491322)
Jiří Čechák (445717)
Jan Ondruch (433341)
Tadeáš Pavlík (487555)
Václav Stehlík (487580)

June 11, 2019

Contents

1 About

Term project for course *PA181 Services - Systems, Modeling and Execution*[1] in year 2019. Within the project, we had to create a fully functional application using the *IBM Cloud*[2] technology including a detailed documentation and a presentation. *doc. Mouzhi Ge, Ph.D.*[3] is the project supervisor.

2 Idea

The core idea was to create and develop a useful and practical application. The application provide services for testing the users in a form of questions and answers. Users are able to test themselves via these questions by selecting the correct answers. There are severals tests in three different types of language (Czech, Slovak and English).

3 Used Technologies

The following technologies were integrated and used during the development process:

- cloud based platform
  - IBM Cloud
- version control system (VCS)
  - GitHub[4]
- continuous integration (CI)
  - Travis CI[5]

Besides the used technologies we were using additional available tools such as:

- IBM Cloud DevOps Toolchain
  - Set or combination of tools that build and deploy the software in a repeatable way with minimal human intervention.
- React
  - A JavaScript library for building user interfaces.
- ASP.NET
  - A framework for building web apps and services with .NET and C#.
- Material-UI
  - React components that implement Google's Material Design.

[1] is.muni.cz/predmet/fi/jaro2019/PA181
[2] cloud.ibm.com
[3] is.muni.cz/auth/osoba/239833
[4] github.com
[5] travis-ci.org

4 Work Division

Our team consisted of 5 members: Adrián Tóth, Jan Ondruch, Jiří Čechák, Tadeáš Pavlík and Václav Stehlík. Each of us was working collaboratively together (except the individual assigned tasks) and the project was done in advance.

Everyone from us was in charge of a certain part of the project. The work was divided as the following:

- Adrián Tóth
  - project initialization (skeleton)
  - VCS initialization
  - Travis CI integration and configuration
  - IBM DevOps toolchain configuration
  - creation of continuous delivery pipeline
  - bug fixing
  - deployment
  - documentation
  - presentation
- Jan Ondruch
  - user research
  - specifications
  - user interface testing
  - usability testing with real users
  - validation and verification
- Jiří Čechák
  - application design (user interface)
  - application frontend
  - backend and frontend interconnection
  - frontend testing
  - bug fixing
- Tadeáš Pavlík
  - analysis
  - specifications
  - application design (idea)
  - user interface testing
  - usability testing with real users
- Václav Stehlík
  - application backend
  - backend and frontend interconnection
  - backend testing
  - troubleshooting
  - bug fixing

5 References

Everything related to the project can be found at:

- github.com/europ/MUNI-FI-PA181

Documentation of the application can be found at:

- github.com/europ/MUNI-FI-PA181/blob/master/doc/doc.pdf

Presentation of the application can be found at:

- github.com/europ/MUNI-FI-PA181/blob/master/pres2/pres.pdf

Source code of the application can be found at:

- github.com/europ/MUNI-FI-PA181/tree/master/src

Setup guide for the application can be found at:

- github.com/europ/MUNI-FI-PA181/wiki/Setup

Domain names of the deployed application are:

- pa181.eu-de.mybluemix.net
- pa181.eu-de.cf.appdomain.cloud

6 Application Initialization

Firstly, we have chosen a stable, reliable and safe platform supporting team project development - GitHub. Furthermore, GitHub provides a version control system management and the integration with IBM Cloud is supported. Subsequently, after the repository was configured properly, we had to choose the technologies. We decided to use #C (general-purpose, multi-paradigm and object oriented programming language) and React (library for building user interfaces) for the project. Also, we decided to use react Material-UI components for our UI. Based on the above, we have initialized the project skeleton - a *'Hello, World!'* cloud foundry application. We took inspiration from *'ASP.NET Core getting started application'*[6] available on GitHub under the terms of *Apache License 2.0*.

7 Application Implementation

The project was implemented via *ASP.NET* framework using a Model–view–controller architectural pattern. The core UI was implemented in JavaScript using React library and Material UI components. This UI is interconnected with backend using our REST API (e.g. the test are passed in JSON format). The application validates the input on the both sides - frontend and backend.

Our application was divided into parts that communicate together and process requests - API, entities, repositories and services. The requests are firstly processed in the API, more precisely endpoint in controller. To process this request, the required actions are delegated to services (the business logic specific for our application) that delegates the related subrequests to repository that provides the data layer communication using wrappers. The application supports also an application log too, which makes the fatal failures easier to find.

[6] github.com/IBM-Cloud/aspnet-core-helloworld

8 Application Deployment

The deployment was configured before the core implementation itself. Deployment was initialized and configured on the already set *'Hello, World!'* application. We wanted to create a fully automated delivery with zero intervention required, which allowed us to concentrate and focus on the code only instead of deployment related set of activities. We were using a continuous delivery approach – automated build and deploy method via IBM's delivery pipeline (more precisely the delivery pipeline stage in the IBM Cloud DevOps toolchain). The delivery pipeline was configured to automatically install all required dependencies, build and deploy the application and also, mark the commit with a success, failure or error status. The development process required to just commit the changes into the version control repository while using a fully automated continuous delivery.

Whole deployment is based on one own properly configured DevOps toolchain. The toolchain is shown in Figure 1, which includes GitHub[7], Travis CI[8] (as other tool) and Delivery Pipeline[9].

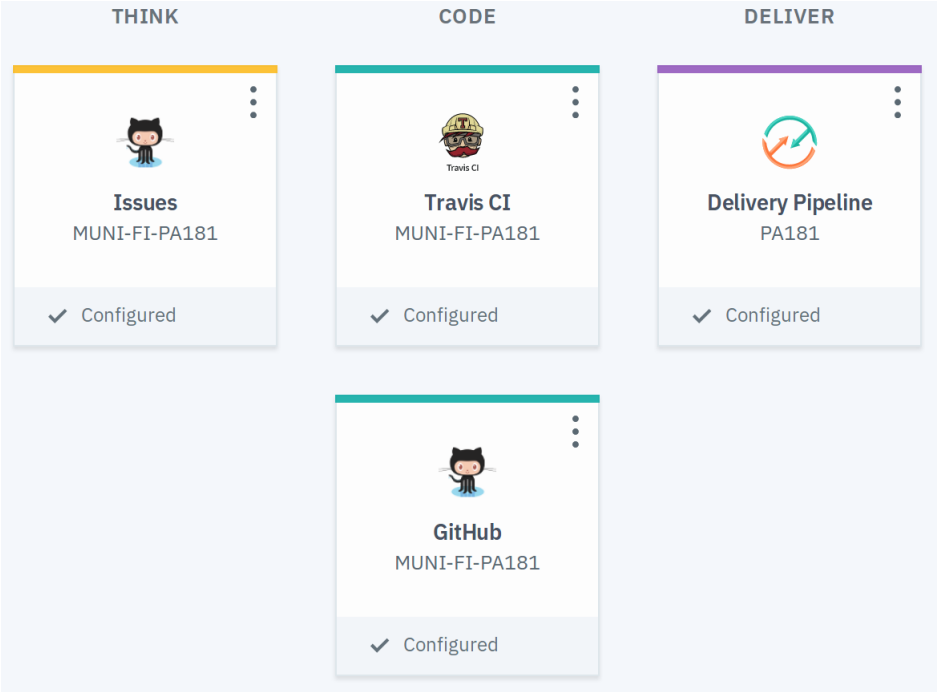

Figure 1: Toolchain

The toolchain providing a continuous delivery service allows us to focus on code only. After the continuous delivery toolchain was set properly, we just had to commit the changes to the repository and the application was immediately built and deployed. As a feedback, the commit was marked with a status based on the build result.

9 Application Instructions

How to launch the application from source can be found in the *setup guide* (see Section 5).

[7] console.bluemix.net/docs/services/ContinuousDelivery/toolchains_integrations.html#github
[8] console.bluemix.net/docs/services/ContinuousDelivery/toolchains_integrations.html#othertool
[9] console.bluemix.net/docs/services/ContinuousDelivery/toolchains_integrations.html#deliverypipeline

10 Use Case Diagram

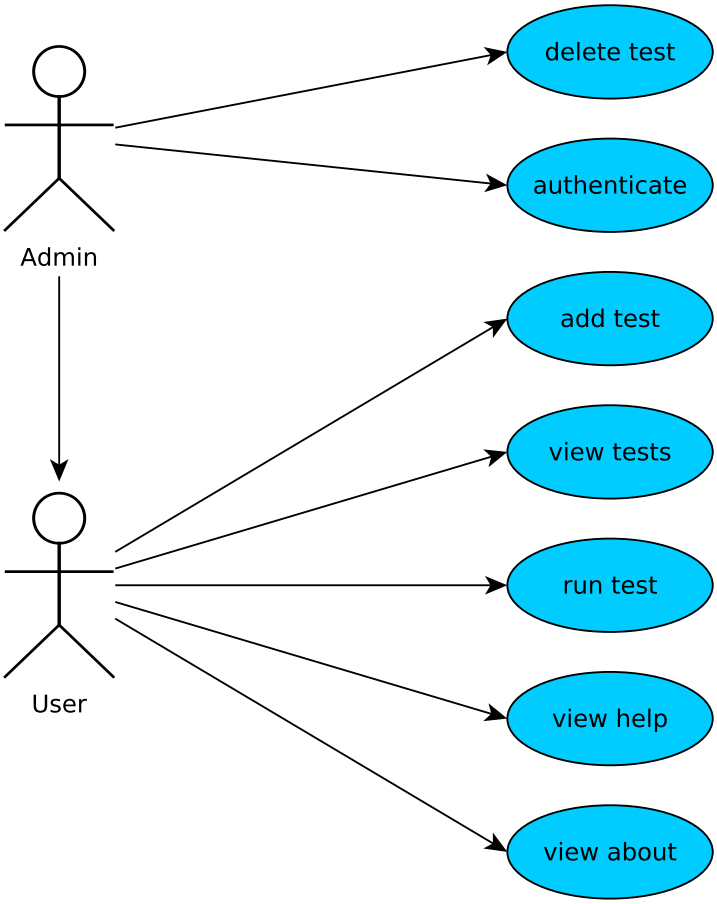

Figure 2: Use case diagram.

11 Flow Diagram

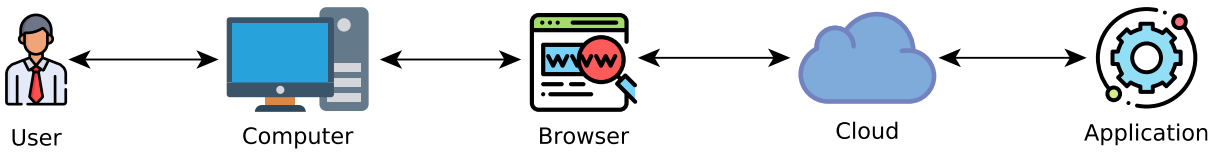

Figure 3: Flow diagram.

12 Screenshots

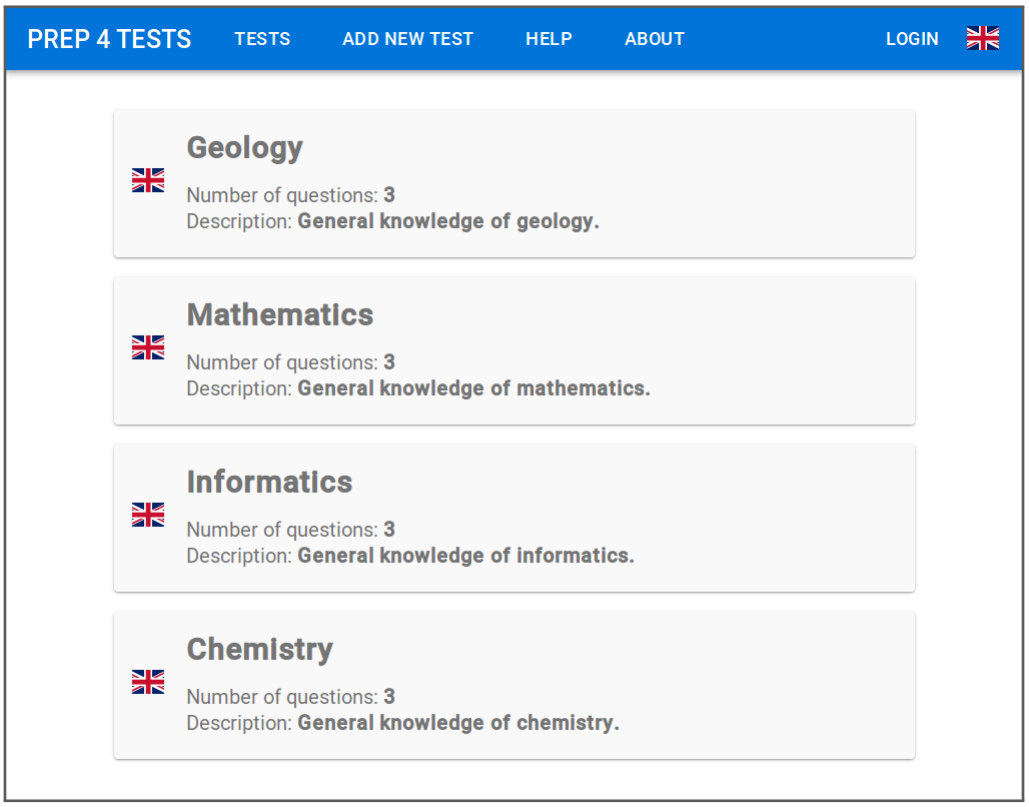

Figure 4: Example 1 - list of available tests.

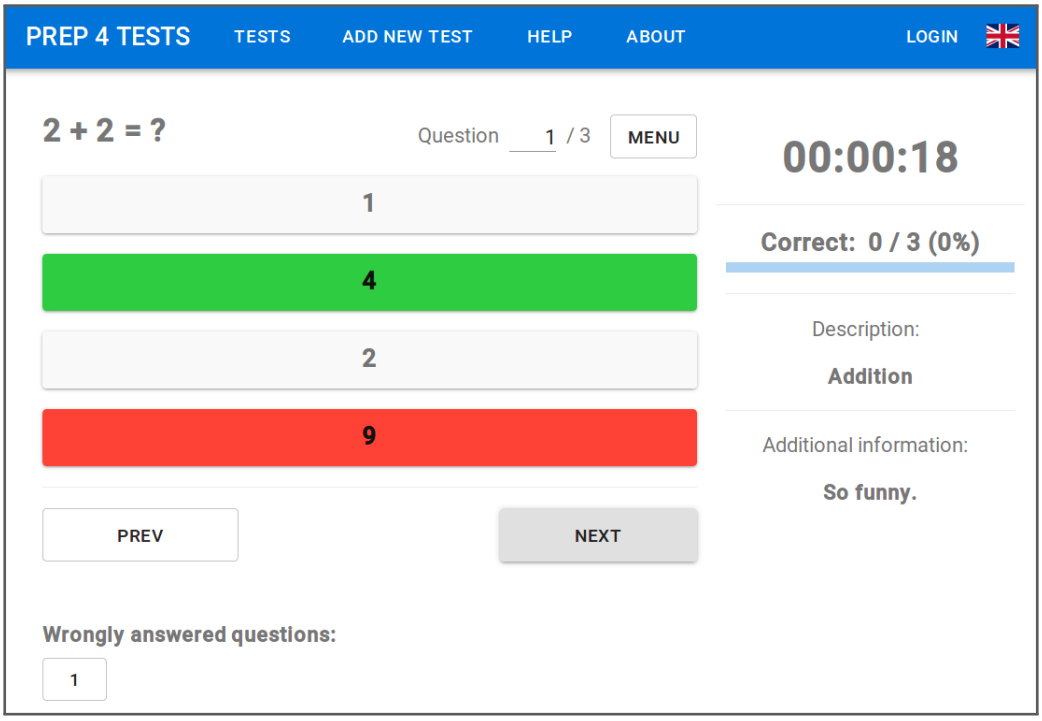

Figure 5: Example 2 - incorrect answer.

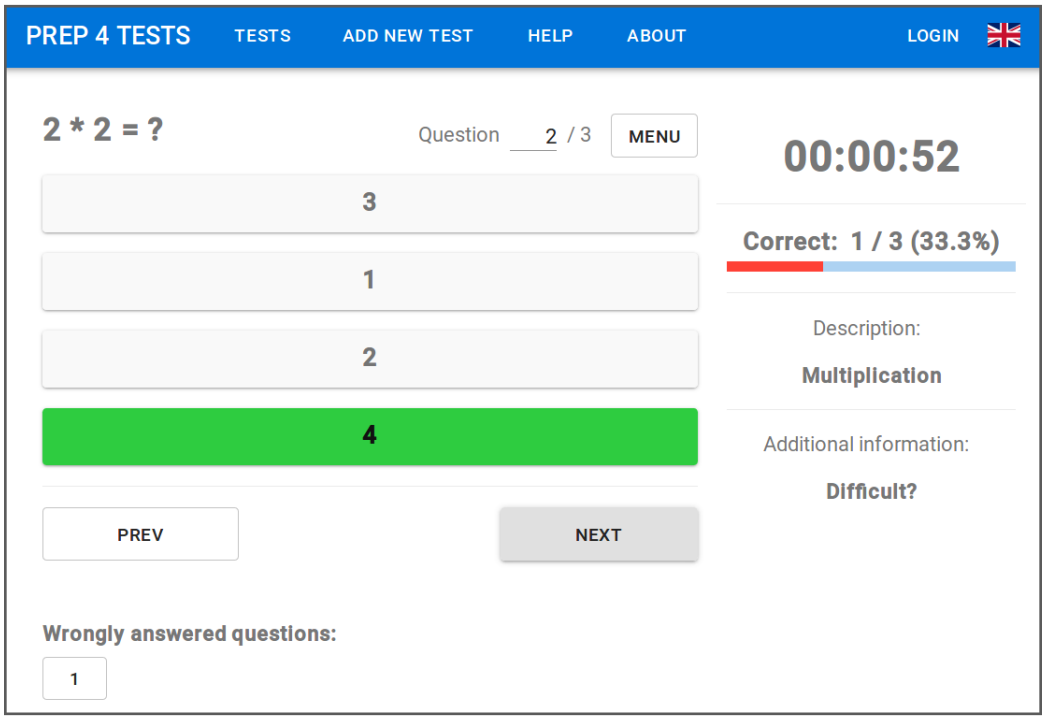

Figure 6: Example 3 - correct answer.

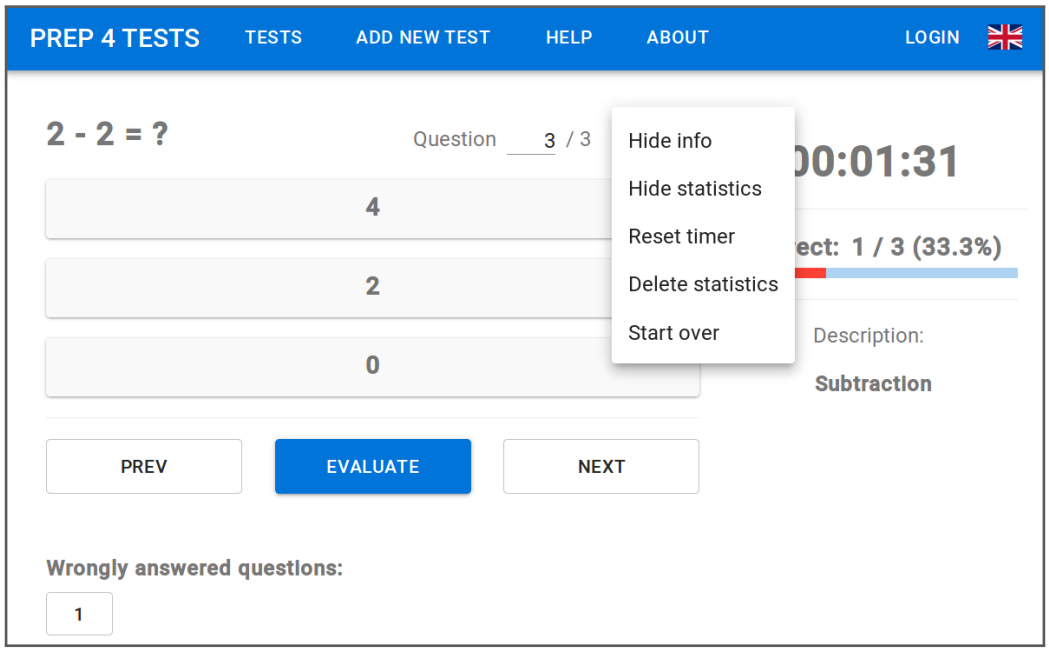

Figure 7: Example 4 - dropdown menu.

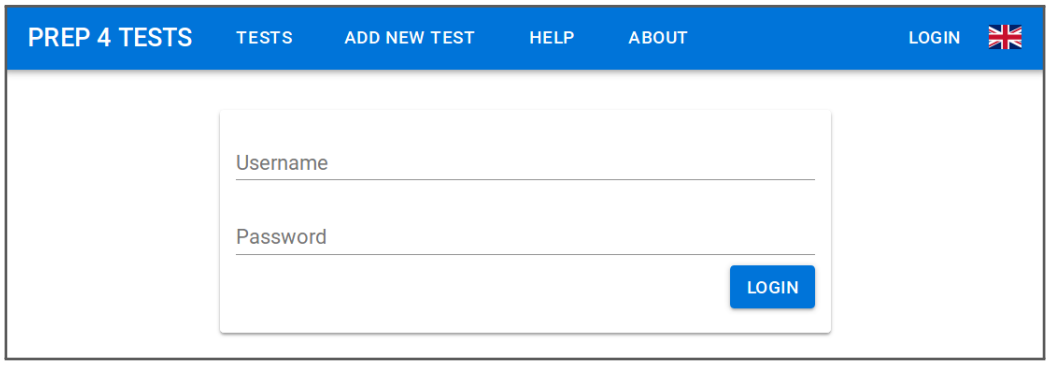

Figure 8: Example 5 - login.

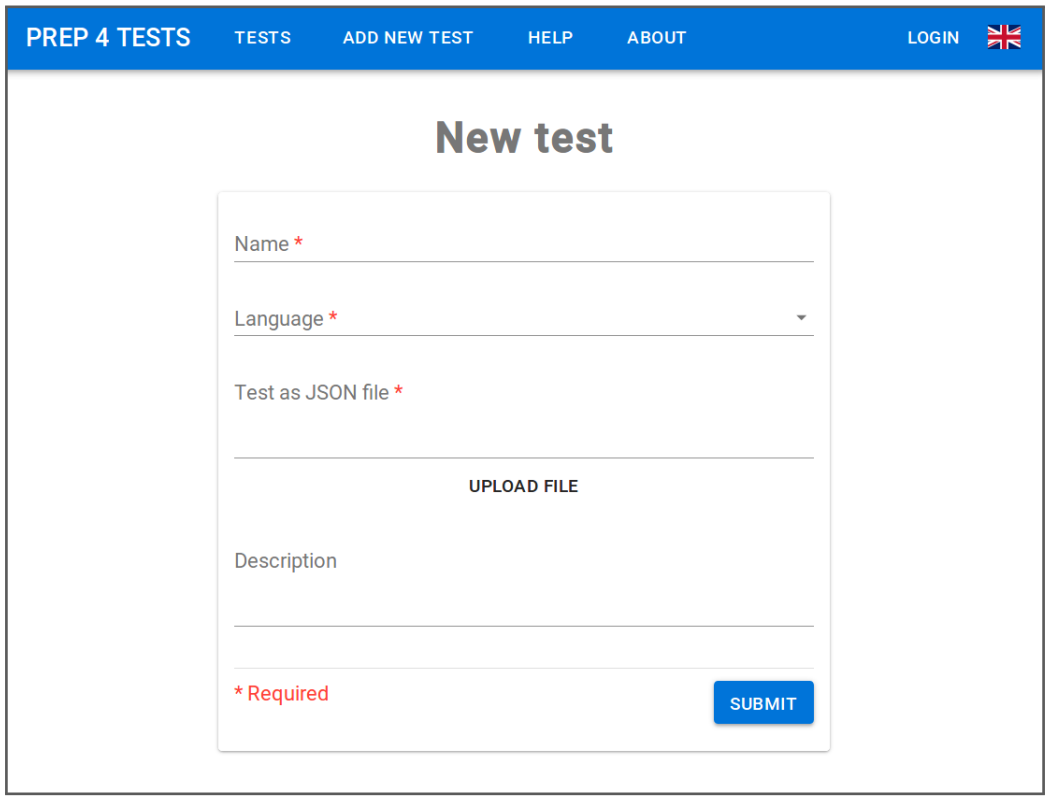

Figure 9: Example 6 - add a new test.

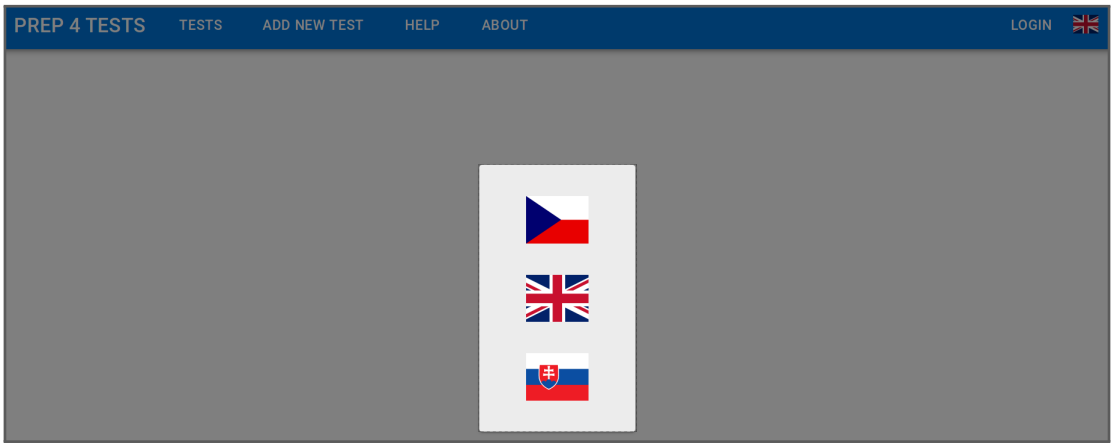

Figure 10: Example 7 - language selection.

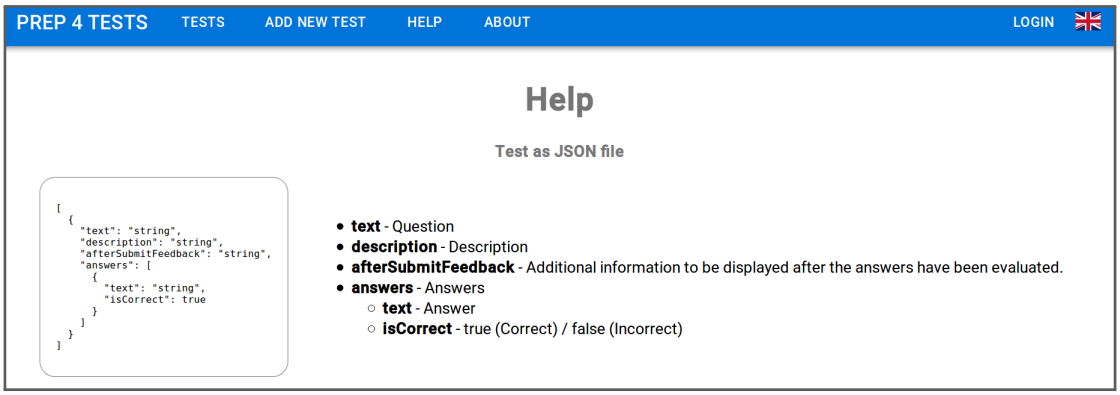

Figure 11: Example 8 - help.

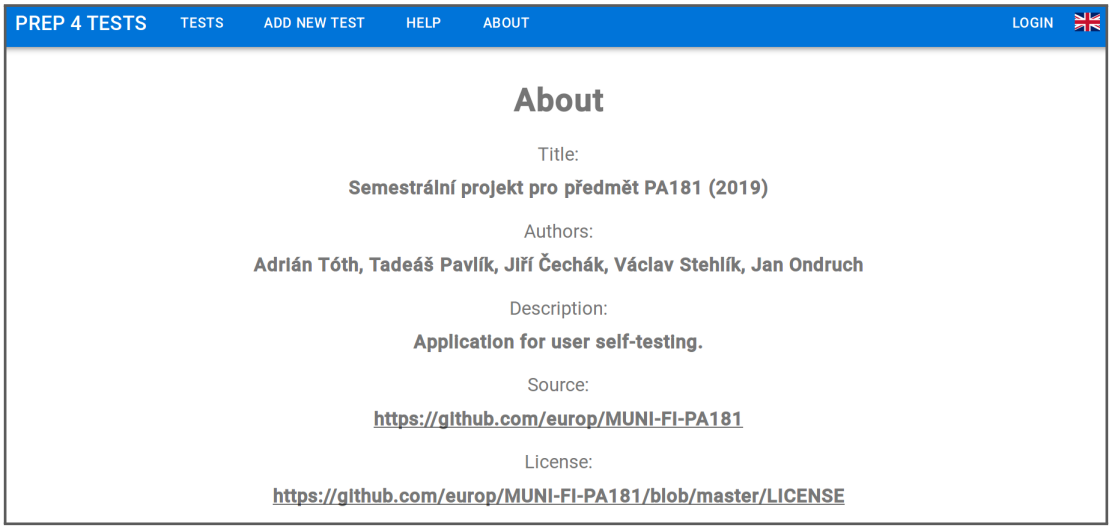

Figure 12: Example 9 - about.

13 Development Issues

During the project development we have faced a few problems that have been reported:

- github.com/IBM-Bluemix-Docs/ContinuousDelivery/issues/13
- github.com/IBM-Cloud/aspnet-core-helloworld/issues/39
  - github.com/IBM-Cloud/aspnet-core-helloworld/pull/40