

# 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)

May 8, 2019

## Contents

1	About	2
2	Idea	2
3	Used Technologies	2
4	Work Division	3
5	References	4
6	Application initialization	4
7	Application implementation	4
8	Application deployment	4
9	Application instructions	5
10	Screenshots	6
11	Development issues	8

# 1 About

Term project for course *PA181 Services - Systems, Modeling and Execution*<sup>1</sup> in year 2019. Within the project, we had to create a fully functional application using the *IBM Cloud*<sup>2</sup> technology including a detailed documentation and a presentation. *doc. Mouzhi Ge, Ph.D.*<sup>3</sup> 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<sup>4</sup>
- continuous integration (CI)
  - Travis CI<sup>5</sup>

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.

---

<sup>1</sup>[is.muni.cz/predmet/fi/jaro2019/PA181](https://is.muni.cz/predmet/fi/jaro2019/PA181)

<sup>2</sup>[cloud.ibm.com](https://cloud.ibm.com)

<sup>3</sup>[is.muni.cz/auth/osoba/239833](https://is.muni.cz/auth/osoba/239833)

<sup>4</sup>[github.com](https://github.com)

<sup>5</sup>[travis-ci.org](https://travis-ci.org)

## 4 Work Division

Our team consisted of 5 members: Adrián Tóth, Jan Ondruch, Jiří Čechák, Tadeáš Pavlík and Václav Stehlík.

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

Domain names of the deployed application are:

- [pa181.eu-de.mybluemix.net](https://pa181.eu-de.mybluemix.net)
- [pa181.eu-de.cf.appdomain.cloud](https://pa181.eu-de.cf.appdomain.cloud)

Source code of the application can be found at:

- [github.com/europ/MUNI-FI-PA181/tree/master/src](https://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](https://github.com/europ/MUNI-FI-PA181/wiki/Setup)

Documentation of the application can be found at:

- [github.com/europ/MUNI-FI-PA181/blob/master/doc/doc.pdf](https://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/pres/pres.pdf](https://github.com/europ/MUNI-FI-PA181/blob/master/pres/pres.pdf)

## 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!'* application.

## 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 database communication using wrappers. The application supports also an application log too, which makes the fatal failures easier to find.

## 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 code only instead of problematic deployment. We were using a continuous delivery approach - automated build

and deploy method via IBM's delivery pipeline. The delivery pipeline was configured to automatically install all required dependencies, build and deploy the application. The development process required to just commit the changes into the version control repository.

Whole deployment is based on one own properly configured DevOps toolchain. The toolchain is shown in Figure 1, which includes GitHub<sup>6</sup>, Travis CI<sup>7</sup> (as other tool) and Delivery Pipeline<sup>8</sup>.

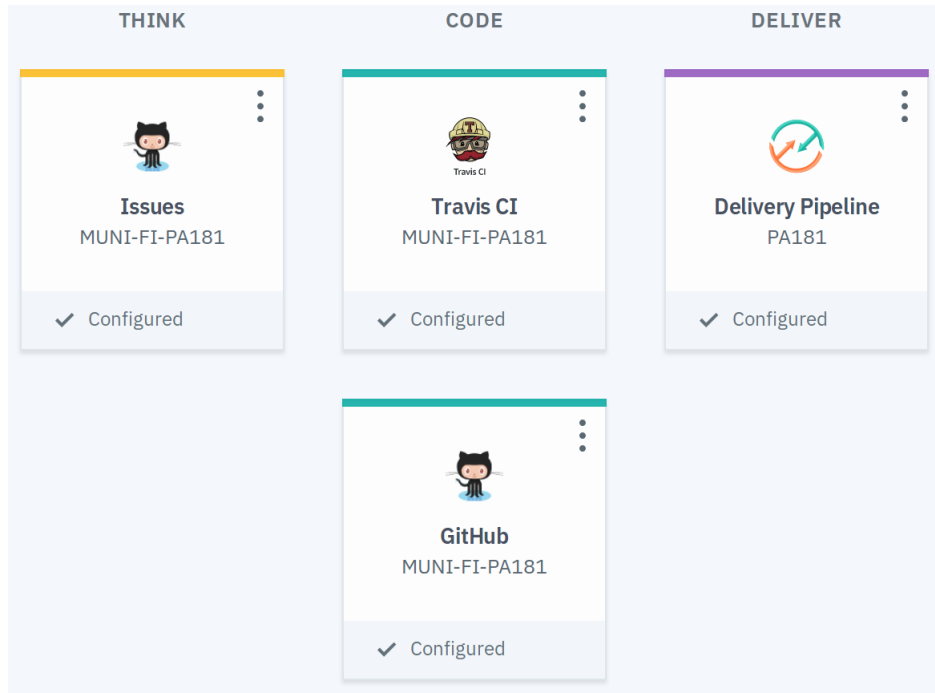


Figure 1: Toolchain

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.

## 9 Application instructions

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

<sup>6</sup>[console.bluemix.net/docs/services/ContinuousDelivery/toolchains\\_integrations.html#github](https://console.bluemix.net/docs/services/ContinuousDelivery/toolchains_integrations.html#github)

<sup>7</sup>[console.bluemix.net/docs/services/ContinuousDelivery/toolchains\\_integrations.html#othertool](https://console.bluemix.net/docs/services/ContinuousDelivery/toolchains_integrations.html#othertool)

<sup>8</sup>[console.bluemix.net/docs/services/ContinuousDelivery/toolchains\\_integrations.html#deliverypipeline](https://console.bluemix.net/docs/services/ContinuousDelivery/toolchains_integrations.html#deliverypipeline)

## 10 Screenshots

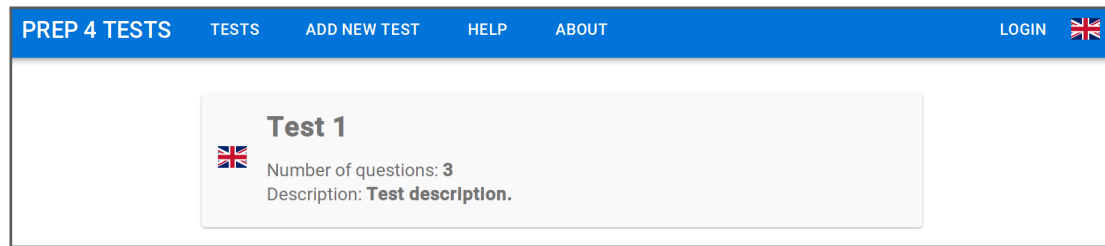


Figure 2: Example 1 - list of available tests.

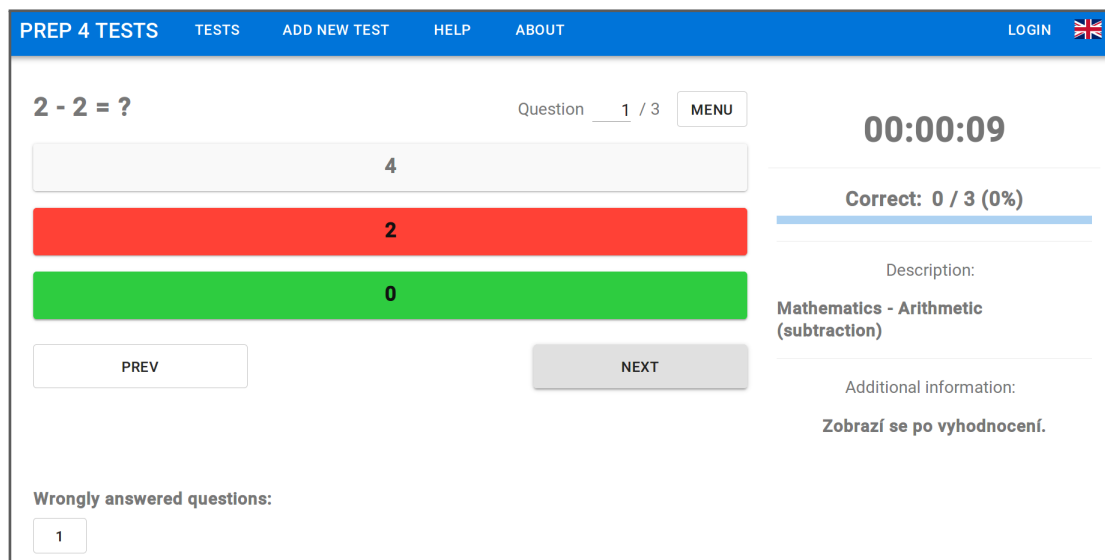


Figure 3: Example 2 - incorrect answer.

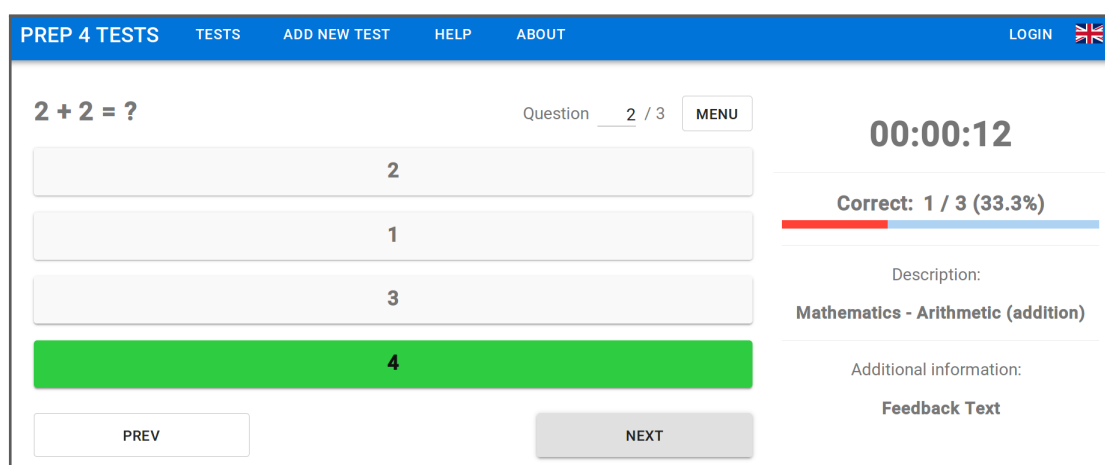




Figure 4: Example 3 - correct answer.



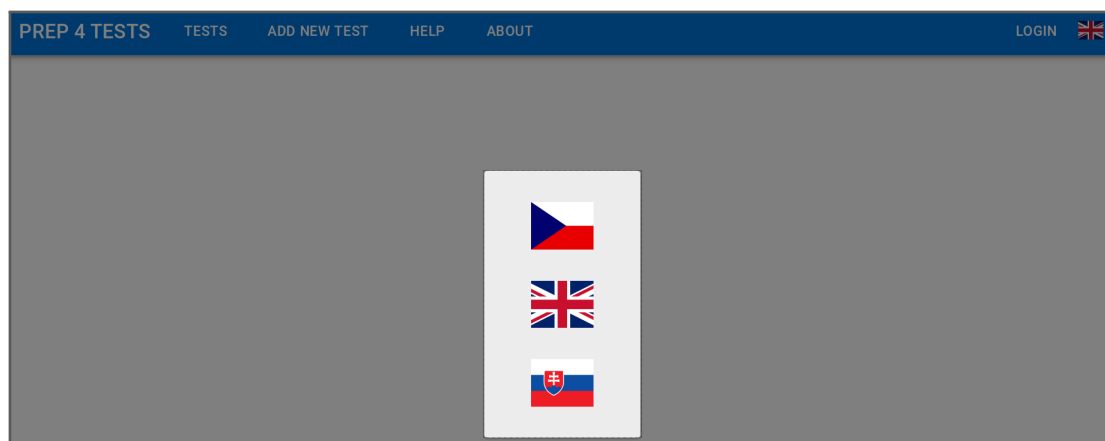
The screenshot shows a web application header with a dark blue bar containing the text 'PREP 4 TESTS' and navigation links 'TESTS', 'ADD NEW TEST', 'HELP', and 'ABOUT'. On the right side of the header are the links 'LOGIN' and a small Union Jack flag icon. The main content area is white and contains a login form with two input fields labeled 'Username' and 'Password'. A blue 'LOGIN' button is positioned to the right of the password field.

Figure 5: Example 4 - login.



The screenshot displays the 'New test' form within the same web application. The header is identical to Figure 5. The form is titled 'New test' and contains several fields: 'Name \*', 'Language \*' (a dropdown menu), 'Test as JSON file \*', an 'UPLOAD FILE' button, and a 'Description' text area. At the bottom of the form, there is a red asterisk followed by the text '\* Required', and two buttons: 'CANCEL' and 'SUBMIT'.

Figure 6: Example 5 - add a new test.



The screenshot shows a language selection modal on a dark gray background. The modal is a light gray rectangle containing three flags stacked vertically: the Czech Republic flag, the United Kingdom flag, and the Russian flag.

Figure 7: Example 6 - language selection.



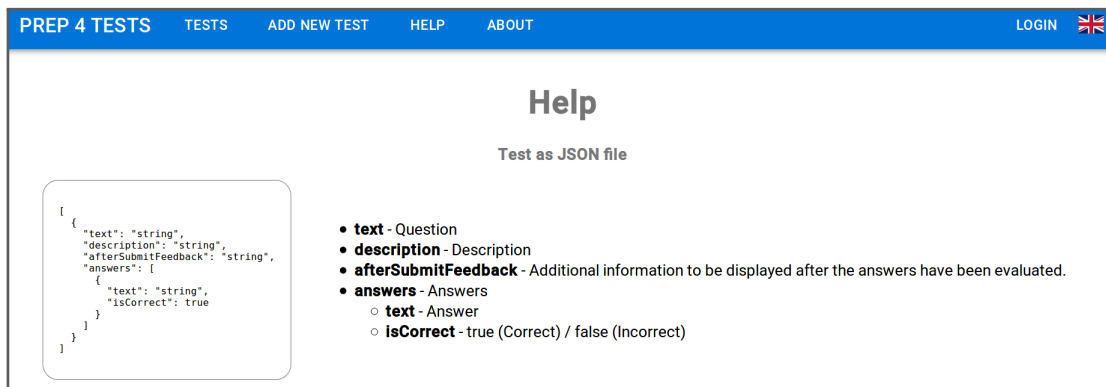


Figure 8: Example 7 - help.

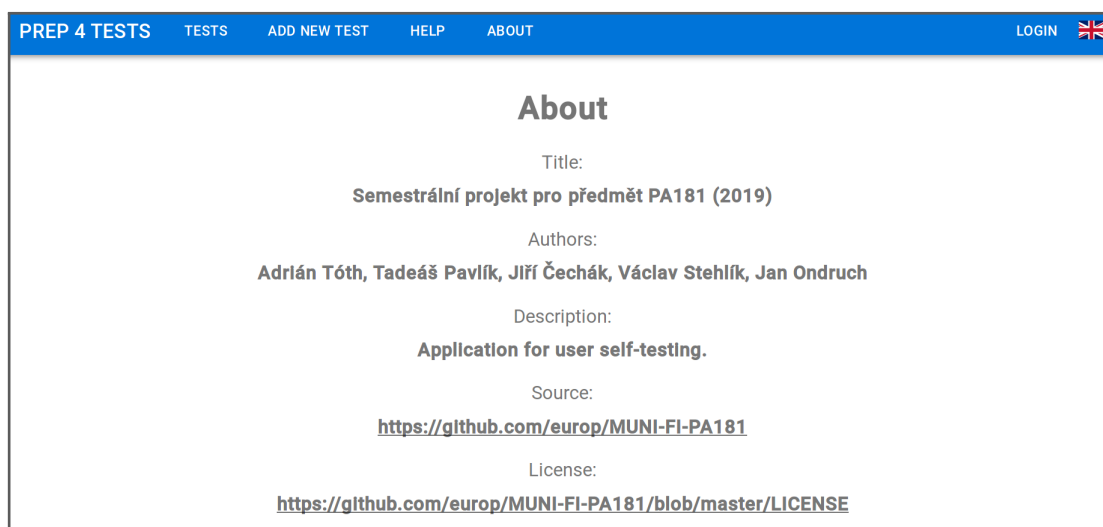


Figure 9: Example 8 - about.

## 11 Development issues

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

- [github.com/IBM-Bluemix-Docs/ContinuousDelivery/issues/13](https://github.com/IBM-Bluemix-Docs/ContinuousDelivery/issues/13)
- [github.com/IBM-Cloud/aspnet-core-helloworld/issues/39](https://github.com/IBM-Cloud/aspnet-core-helloworld/issues/39)