SE 3XA3: Software Requirements
Spann

Team 5
Christopher Stokes — stokescd
Varun Hooda — hoodav

December 7, 2016

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	Version	Notes
Oct. 1, 2016	1.0	Initial Changes
Oct. 11, 2016	1.1	Finishing Changes
Dec. 6, 2016	2.0	Feedback changes

1 Project Drivers

1.1 The Purpose of the Project

The purpose of the project is to develop a web browser based Python IDE application. The application will provide an environment similar to desktop based integrated development environments, but with the convenience of a seamless experience regardless of their operating system and hardware platform.

1.2 The Stakeholders

1.2.1 The Client

The client for whom this application is being developed is Dr. Smith, the professor of Software Engineering 3XA3.

1.2.2 The Customers

The customers of the application will be python developers looking for a convenient platform that allows them to develop from almost anywhere, on almost anything, as long as they have an internet connection.

1.2.3 Other Stakeholders

Other stakeholders include students and teachers looking for a platform to learn and teach fundamentals of programming without the trouble of setting up a programming environment by themselves.

1.3 Mandated Constraints

The project needs to run on software and hardware that McMaster University has accesses to and has the licenses for.

The project's front end UI needs to execute in a modern web browser. The server, which serves the front end UI by means HTTP, will require a recent version of the Windows® operating system.

1.4 Naming Conventions and Terminology

The following terms may be used throughout this document:

Us/We The members of the group who are developing this project.

Libre Free and Open Source Software. Specifically, free in terms of freedom as appose to price; thus freedom to modify, view and possibly redistribute the software.

Application The Spann Online IDE application.

IDE Integrated Development Environment is a application for developing software within a standalone suite of tools.

SaaS (Software as a Service) A model of computation in which a piece of software if provided to the user as a service rather than a product.

Virtualization A method of computation in which a software utility isolates a part of the computational space from the rest of the system to increase modularity and security of the system as a whole.

HTML5 Fifth version of the Hyper Text Markup Language.

HTTP Hyper Text Transfer Protocol.

UI User Interface

Front end The user facing part of the application.

Back end The part of the application that serves and services the front end.

SQL Standard Query Language.

API Application Programming Interface.

1.5 Relevant Facts and Assumptions

The application will assume the user has a modern, HTML5 compatible browser that has JavaScript enabled. The application will be tested to ensure it is functional on the major modern browsers (see appendix).

The application assumes the users will input data into the application using a combination of the mouse and keyboard. Mouse inputs will be used to

perform various actions (navigation, button functionality) and the keyboard input will be used to get textual input from the user. The user is assumed to have full functionality of the mouse and keyboard. The user is also assumed to be able to enter any printable character input into any of the application's input fields or text areas. Similarly, the user is assumed to have the capability of clicking anywhere on the application's front end UI.

2 Functional Requirements

2.1 The Scope of the Work and the Product

The scope of the work is limited to the supporting documentation, the code to implement the application's functional and non functional requirements, and the supporting tests to test the application.

2.1.1 The Context of the Work

The work will be mainly software design, implementation using the chosen programming languages and testing to ensure the application works as specified.

2.1.2 Work Partitioning

In general the split of work is approximately equal in client and server side code. The difference in who is responsible for the underlining architecture development and who is responsible for the application specific development. In general the underlining setup and architecture development is handled by Christopher Stokes and the application specific development by Varun Hooda.

Christopher Stokes:

- Project and code design
- Database design
- SQL code generation framework
- UI framework design
- UI design

- Testing

Varun Hooda:

- Database design
- UI design and development
- API development
- Client and server side algorithm design and development
- Testing

2.1.3 Individual Product Use Cases

Algorithm Testing: A main use case of this project is to be able to design individual parts of larger projects or algorithms without it interacting with the larger project. Because of how quick this project is a creating projects it makes it incredibly easy to test small sections of code during development. This can be done in a project or in the console.

Full Project Development: This project can be used to develop full projects with virtual know limitations on what can be supported. There are a number of reason this would be done, such as it means the developer does not need to create a local environment and is able to develop on any deceive with access to the web.

2.2 Functional Requirements

Mode The application shall have two modes of operation U and O (see appendix).

Editing The application shall allow the user to edit and save new and existing files (U mode).

Editing Support The application shall provide the user with language specific code highlighting while the user is editing a file (U or O).

File Handling The application shall manage (view, store, track) files created by the user (U).

Code execution The application shall allow the user to execute a file containing python code and the application will display the output of the execution to the user (U or O).

Shell Interpreter The application shall provide the user with a interactive shell for executing individual commands (specifically python) and present the user with the output of the command (U or O).

Networking The application shall execute the code and store user created data (file, metadata, user account data) on the server and forward the output and other data to the browser via a network connection.

Accounts The application shall allow new users to create an account and allow existing users to login into their user account (O).

Account Mangement The application shall allow the user to manage their accounts through a user interface on the application (manage account details such as password, email, etc.) (U).

3 Non-functional Requirements

3.1 Look and Feel Requirements

This application needs to maintain a unified enterprise look and feel. A major goal of this application is to increase development speed, this means it is incredibly important that the user experience is clean. To achieve this, options such as save, properties, and right-click must be constant throughout the application. As a target, this application must respond and feel like a desktop application while running in browser meaning all actions that one would expect to work in a desktop IDE such as Eclipse must work in browser.

3.2 Usability and Humanity Requirements

Users of this project are generally of a technical background and the nature of the work is technical. This means the product is able to be designed in a more technical nature but it is still important that the product is usable by ensuring a consistent design and button location across all screens.

3.3 Performance Requirements

3.3.1 Client

The client side UI shall remain responsive while performing actions on behalf of the user (processing input, communicating with server, displaying output). Hence, it is most important that the application can run asynchronously at all times to not lock up the browser.

3.4 Operational and Environmental Requirements

The environmental impact of an engineering project should always go under considering when designing and implementing a project. For this project we aim to reduce the hardware requirements required to use the application; Thus ensuring the application is as portable as possible and functions on a wide range of hardware.

Platform System Requirements The application shall employ efficient algorithms, and performance boosting optimization techniques implemented in the application code and the browser on which the application executes.

3.5 Maintainability and Support Requirements

It is highly important that the source for this project is maintainable as the features of the IDE will grow as more advanced tools are added to encompass or use cases and parts of the development processes. Furthermore web APIs and support changes incredibly fast so it is important that the code stays up to date to support the best technologies.

3.6 Security Requirements

This application should provide a secure platform and carry out its functionality in a secure manner. This means the application needs to meet the following a set of security requirements:

- The application shall only allow the correctly authenticated user access the project files associated with the account being authenticated.

- The application executes user code in secure manner, isolated from the rest of the system to ensure any malicious code doesn't compromise the security of the system and application.

3.7 Cultural Requirements

The project may need to be translated or support translation if a significant number of users' primary language is not English.

3.8 Legal Requirements

The project needs to abide by all international and domestic laws, as well as all of McMaster University's Policies, Procedures and Guidelines. Depending on the license under which this project is released we may also need to consider other legal obligations (this is will made clear once a license has been chosen).

3.9 Health and Safety Requirements

The project will attempt to be as ergonomic and health conscious as possible. Thus the following will be incorporated into the project:

Minimal User Interface Allow the user to do as much as possible using as little work (clicking, navigating the UI, scrolling) as possible.

This section is not in the original Volere template, but health and safety are issues that should be considered for every engineering project.

4 Project Issues

4.1 Open Issues

The primary issue we mean to address with the product is the lack of online fully functional IDE application for the Python programming language. This application should be accessible through a modern web browser and be presented to the user through the SaaS (Software as a Service) model.

4.2 Off-the-Shelf Solutions

The are various off-the-shelf solutions available for writing and running code on the web browser, but these solutions have little to no support for the python programming language besides being able to provide simple code completion and ability to execute the code. Thus, for this project, we're aiming to both support execution of python code, code completion and add IDE-like, developer friendly features (refactoring support and file management).

4.3 New Problems

The follow new problems have arisen during the design and development of this project,

- Handling python code execution in a secure manner to security the implications that are present in the execution of unknown code.
- Maintaining compatibility between multiple browsers and their individual web technology implementations.

4.4 Tasks

The tasks for this project are as follows,

- Design the server and the front end
- Design the database schema
- Begin implementing the front end
- Begin implementing the database
- Begin implementing the server
- Integrate the front end and server (networking)
- Test the system, ensure functionality and security

4.5 Migration to the New Product

The process of migrating to the new product will not require the user to go through any complicated procedure. The process will be signing up for the new application, copying the code from the old product or writing new code on the new application.

4.6 Risks

One inherent risk with allowing users to execute code on your servers is the user's ability to perform malicious actions. This can result in damage to the hardware, the software stack and to the data on the server. Another risk is the possibility of some fault in the system causing user's to lose data or the project to lose business critical data or damage the hardware or software stack.

4.7 Costs

The project will be mainly using libre software that is available without costs, as well as non-libre software this is available to us without cost. If the platform is to be scaled for public usage, the project will need to be hosted on some server (or multiple server depending on rate of user adoption) which would have a monetary cost.

4.8 User Documentation and Training

The project will be fully documented, including design documents, testing documents, well commented code. The documentation will also include resources for new users – tutorials, guides and user manual. This level of documentation should hopefully provide users (as well as developers) enough material to user (and perhaps contribute to) the application.

4.9 Waiting Room

The list of potential features we hope to implement in the future is,

- Moving the application platform from dedicated hardware to a online dynamic hosting provider to allow the application to scale to a large number of concurrent users.

- Support for other languages besides python
- Support for large, multi-team, distributed projects

4.10 Ideas for Solutions

The project members will continue to develop new solutions to the issues presented in this section. Some of the potential solutions are,

Secure code execution Using a virtualization software, the code can be executed in a secure manner in an isolated environment on the same hardware as the application server.

Reducing power usage Using an online dynamic hosting service (such as Amazon Web Services) we can ensure our application does not require dedicated, fixed hardware. This method allows the application to dynamically allocate computational resources based on current usage levels, thus reducing redundant, unused computational power and reducing power consumption.

5 Appendix

5.1 Supported Web Browsers

- Mozilla Firefox
- Google Chrome
- Microsoft Edge

5.2 Symbolic Parameters

The application shall have two Modes of operation:

(U)ser mode The user has signed into an account.

(O)pen mode The user has not signed in to an account.