Web System Engineering

**Project Final Report**

|  |
| --- |
| **Team : <name>** |
| List of team members |

**I. Project Idea**

|  |
| --- |
| **Project Pitch** |
| Define the idea of your project in one sentence |
| The “Transaction management system” (TMS) is a web application which helps people to find and change information about their accounts, perform transactions (transfers and payments), view their accounts summary based on each user, easy to use from any web connection and for anyone who already has an account. |

|  |
| --- |
| **Project Detailed Description** |
| List and describe in detail your project idea, settings and technical and usage restrictions, limitations (provide sufficient details). |
| This system will consist of a web portal, used for managing the information about the users accounts and the system as a whole.  Since this is a data-centric product, it will need somewhere to store the data. For that, a database will be used. The web portal will communicate with the database, being able to add and modify data. All of the database communication will go over the Internet.  The web portal will provide the following functionalities:   * User Registration * User Login * User Logout * Password recovery * Personal and account information update * Money transfer * Transactions history * Payments * Customer services * Loan simulator * Currency converter  User Classes and Characteristics There will be three types of users who can use the application:   * The admin –has already an account and has access to the list of clients, has the right to activate/deactivate an account and can reset a password at user’s request. * The unregistered user – can see information about the application and register/create a new account. * The registered user –can use all the functionalities listed in above section. * **Specific requirements**   This section contains all of the functional and quality requirements of the system. It gives a detailed description of the system and all its features.   * + User interfaces   The user of the web application should see the log-in page when they open the application. If the user has not registered, he/she should be able to do that on the log-in page. Here the user fills in their email address as well as a password. The required information in order to configure the user’s profile shall be updated after register phase.  If the user is not a first-time user, he/she should be able to sign in with their email and password .  Every user shall have a home page where they can see the balance of their accounts, and their card limits.  Users shall be able to transfer amount of money to other accounts of their own, or to other users accounts using the selected currency. If the data entered is valid, the operation will succeed .  Every user shall choose from the services tab to pay various bills, services such as phone credit, car vignette .  The account and personal information such as first and last name, address, password, email address, phone number, bank accounts IDs, credit limits shall be configured at any moment the user want, by navigating to the Profile view and choosing between the 2 tabs .  The transactions history shall be easily viewed for a maximum period of 2 months with details regarding every transfer in or out of the account(s) |

**II. Project Technical Solution**

|  |
| --- |
| **Web Technologies** |
| Describe the technical solution of your project, particularly, depict an architecture, list and describe technical components and interactions among them, used frameworks, databases, servers, etc. |
| This system will consist of a web portal, used for managing the information about the users accounts and the system as a whole.  Since this is a data-centric product, it will need somewhere to store the data. For that, a database will be used. The web portal will communicate with the database, being able to add and modify data. All of the database communication will go over the Internet.    Figure Block Diagram Operating Environment The application can be accessed on every web browser that runs on any computer operating system. Design and Implementation Constraints The Internet connection is a constraint for the application. Since the application fetches data from the database over the Internet, it is crucial that there is an Internet connection for the application to function.  The web portal will be constrained by the capacity of the database. If the database size becomes too large, the application may be forced to queue incoming requests and therefor increase the time it takes to fetch data.  The application does not provide mobile functionalities. It can only be accessed on a PC that has a web browser and an Internet connection .  The used language will be English only, since is widely used and known by the most of the population. Assumptions and Dependencies One assumption about the product is that it will always be used on web browsers that have a proper Internet network connection.  The UI is dependent on the web browser it runs on.  Operating System Environment needed to run the aplication at the best functional level: Windows10 Pro, Version 1709 and later editions or versions  Web Browser needed to run the aplication at the best functional level : Chrome 65 and later versions. Communications Interfaces The .NET Framework that is used in developing the application provides comprehensive support for the *HTTP* protocol, which makes up the majority of all Internet traffic. Technologies Used The system is created using the Asp .NET technology and the Entity Framework. ASP.NET Core is a new open-source and cross-platform .NET framework for building modern cloud-based web applications, is a mature web platform that provides all the services that you require to build enterprise-class server-based web applications using .NET on Windows. To have an organized code I used the MVC(Model-View-Controller) pattern which gives a powerful, patterns-based way to build dynamic websites that enables a clean separation of concerns and that gives you full control over markup for enjoyable, agile development. ASP.NET MVC includes many features that enable fast, TDD-friendly development for creating sophisticated applications that use the latest web standards. ASP.NET MVC is an open-source software from Microsoft. Its web development framework combines the features of MVC (Model-View-Controller) architecture, the most up-to-date ideas and techniques from Agile development and the best parts of the existing ASP.NET platform.  To create the interface I used Razor which is based on ASP.NET, and designed for creating web applications. It has the power of traditional ASP.NET markup, but it is easier to use, and easier to learn. Razor is a markup syntax that lets you embed server-based code into web pages using C# and VB.Net. It is not a programming language. It is a server-side markup language. Razor has no ties to ASP.NET MVC because Razor is a general-purpose templating engine. You can use it anywhere to generate output like HTML. It's just that ASP.NET MVC has implemented a view engine that allows us to use Razor inside of an MVC application to produce HTML. You will have a template file that's a mix of some literal text and some blocks of code. You combine that template with some data or a specific model where the template specifies where the data is supposed to appear, and then you execute the template to generate your output. The razor code is placed in the views files, they are controlled by the controllers commands and takes the data from the models.  Entity Framework is an open-source ORM framework  for .NET applications supported by Microsoft. It enables developers to work with data using objects of domain specific classes without focusing on the underlying database tables and columns where this data is stored. With the Entity Framework, developers can work at a higher level of abstraction when they deal with data and can create and maintain data-oriented applications with less code compared with traditional applications.  Official Definition: “Entity Framework is an object-relational mapper (O/RM) that enables .NET developers to work with a database using .NET objects. It eliminates the need for most of the data-access code that developers usually need to write.” |

**III. Project Results**

|  |
| --- |
| **Project Requirements Fulfillment** |
| Describe your final implementation status according to your requirements. Justify the unfulfilled requirements (if any). |
| The implementation is about 80% in concordance with initial design.  Drawbacks:  Currency converter is not implemented.  Mail sending not functional. |

|  |
| --- |
| **User Interfaces** |
| Describe the user interfaces (put screenshots if necessary).    Register view    Login View |
| My accounts view    New account view    Transfer to my accounts view    Transfer to other accounts view    My transfers view    Pay a bill view    Pay tax view    Transaction history view    FAQ view    Map view    Admin home view    Admin activate account view    Admin deactivate account view |

|  |
| --- |
| **Project Adaptation** |
| Describe any required adaptation during your project implementation. For example, additional features which are not included in the initial requirements. |
|  |

|  |
| --- |
| **Project Showcase** |
| Put public URL of your project. |
|  |

**IV. Project Future Potential**

|  |
| --- |
| **Potential Future Work** |
| Describe how you would progress with and/or improve your project, if you are to continue working on it. |
| Security updates needed.  Password change option desirable. |