**1.Development**

**1.1.Controllers**

**1.1.1 Account Controller**

Handles any request which involves altering the state of an account

**1.1.1.1 public ActionResult Login()**

Returns the login page

**1.1.1.2 public ActionResult Profile()**

Returns the profile page

**1.1.1.3 public ActionResult Grades()**

Returns the grade page

**1.1.1.4 public ActionResult Settings()**

Returns the settings page

**1.1.1.5 public ActionResult Login(string username, string password)**

Handles a post event and saves in the current session the id,username, role and the the current state of the login.If the username and password don’t match, the state of the login is false else it is true

**1.1.1.6 public ActionResult Logout()**

Handles the logout and changes the state of the login to false

**1.1.1.7 public ActionResult ChangePassword(ChangePasswordViewModel model)**

Handles the post event and changes the password to the current logged in user

**1.1.1.8 public ActionResult EditProfile(EditProfileViewModel model)**

Handles the post event and updates the current information of the logged in user according with the model passed as parameter

**1.1.1.9 public bool validateCNP(long cnp)**

Helping function which checks if a CNP has the correct format

**1.1.2.0 public bool validateEmail(string email)**

Helping function which checks if an email has the correct format

**1.1.2 Management controller**

Handles any request which involved adding/deleting users, adding/deleting project and assigning projects to users

**1.1.2.1 public ActionResult Index()**

Returns the management page

**1.1.2.2 public ActionResult AddUser(string username, string password, string firstName, string lastName, long cnp, string email, string role)**

Post event which adds an user to the database

**1.1.2.3 public ActionResult DeleteUser(string username)**

Post event which deletes an user from the database, which has the same username as the parameter

**1.1.2.4 public ActionResult AddProject(string projectName, bool? projectOptional)**

Post event which adds a new project to the database

**1.1.2.5 public ActionResult DeleteProject(string projectName)**

Post event which deletes a project from the database

**1.1.2.6 public ActionResult SetProject(string username, string projectName)**

Post event which assigns a project to an user

**1.1.3 Project controller**

Handles any event which involve project requests

**1.1.3.1 public ActionResult Index()**

Returns the projects page

**1.1.3.2 public ActionResult ViewProject(int ID)**

Returns the information about a specific project given by the parameter ID specified

**1.1.3.3 public ActionResult SaveInformation(string description, DateTime deadline, int ID)**

Saves the specified information to the project given by the ID parameter

**1.1.3.4 public ActionResult ViewAssigment(int ID)**

View the assignment denoted by the parameter ID

**1.1.3.5 public ActionResult SubmitComment(string comment, int userID, int ID)**

Submits a comment to the given project ID by the user with the ID userID

**1.1.4 SubmitProject Controller**

Handles the download/upload file events

**1.1.4.1 public ActionResult SubmitProject(SubmitModel submitModel, int projectId, int userId)**

Handles the file upload to a specific project

**1.1.4.2 public ActionResult DownloadProject(int projectId, int userId)**

Handles the file download to a specific project

**1.2 Models**

**1.2.1 Account View Models**

**1.2.1.1 Change password view model**

**public class ChangePasswordViewModel**

**{**

**public string OldPassword { get; set; }**

**public string NewPassword { get; set; }**

**public string ConfirmPassword { get; set; }**

**}**

**OldPassword** – old password of the user

**NewPassword** – new password of the user

**ConfirmPassword** – should be the same password as the **NewPassword**

**1.2.1.2 Edit profile view model**

**public class EditProfileViewModel**

**{**

**public string FirstName { get; set; }**

**public string LastName { get; set; }**

**public long CNP { get; set; }**

**public string Email { get; set; }**

**}**

**FirstName** – user’s first name

**LastName** – user’s last name

**CNP** – user’s cnp

**Email** – user’s email

**1.2.2 Project Model**

**1.2.2.1 Project**

**public class Project**

**{**

**public int ID { get; set; }**

**public string Name { get; set; }**

**public string Description { get; set; }**

**public DateTime Deadline { get; set; }**

**public bool Optional { get; set; }**

**}**

**ID** – project’s model primary key

**Name** – project’s name

**Description** – project’s description

**Deadline** – project’s date deadline

**Optional** – boolean which marks if the project is optional or not

**1.2.3 Submit model**

**1.2.3.1 SubmitModel**

**public class SubmitModel**

**{**

**public HttpPostedFileBase File { get; set; }**

**}**

**File** – file which is uploaded or downloaded

**1.2.4 User Model**

**1.2.4.1 User**

**public class User**

**{**

**public int ID { get; set; }**

**public string Username { get; set; }**

**public string Password { get; set; }**

**public string FirstName { get; set; }**

**public string LastName { get; set; }**

**public long CNP { get; set; }**

**public string Email { get; set; }**

**public static bool LoggedIn()**

**{**

**return (bool)HttpContext.Current.Session["Logged"];**

**}**

**public static string getName()**

**{**

**return (string)HttpContext.Current.Session["Name"];**

**}**

**public static int? getID()**

**{**

**return (int)HttpContext.Current.Session["ID"];**

**}**

**};**

**ID** – primary key

**Username** – user’s username

**Password** – user’s password

**FirstName** – user’s first name

**LastName** – user’s last name

**CNP** – user’s cnp

**Email** – user’s email

**LoggedIn()** – returns if the user is logged in or not

**getName()** – returns name of the logged in user

**getID()** – returns id of the logged in user

**1.2.5 User Role**

**1.2.5.1 UserRole**

**public class UserRole**

**{**

**public int UserID { get; set; }**

**public roles Role { get; set; }**

**public enum roles**

**{**

**student,**

**professor,**

**administrator**

**}**

**}**

**UserID** – primary key for the current table and foreign key to the user’s id

**Role** – user’s role

**1.2.6 User Project many to many relationship**

**1.2.6.1 Users\_Projects**

**public class Users\_Projects**

**{**

**public int UsersProjectsID { get; set; }**

**public int ProjectID { get; set; }**

**public int Mark { get; set; }**

**public string FileName { get; set; }**

**public string FileExtension { get; set; }**

**public Byte[] File { get; set; }**

**}**

**UsersProjectID** – primary key

**ProjectID** – foreign key to the project table

**Mark** – user’s mark to the project

**FileName** – file’s name

**FileExtension** – file’s extension

**File** – current file in the project

**1.2.6.2 Users Project Comments**

**public class Users\_Projects\_Comments**

**{**

**public int ID { get; set; }**

**public int UsersProjectsID { get; set; }**

**public int UserID { get; set; }**

**public string Comment { get; set; }**

**}**

**ID** – primary key

**UsersProjectID** – foreign key to the many to many relationship

**UserID** – foreign key to the user’s id

**Comment** – comment in the project

**2.Tests**

**2.1.1 public void AddUserTest()**

Tests if the user is added correctly

**2.1.2 public void TestUserUnique()**

Tests if the user is unique

**2.1.3 public void LoginFailedTest()**

Tests if you don’t login

**2.1.4 public void DeleteUser()**

Tests if the user is deleted correctly

**2.1.5 public void AddProjectTest()**

Tests if the project is added correctly

**2.1.6 public void DeleteProjectTest()**

Tests if the project is deleted correctly

**2.1.7 public void SetProjectTest()**

Tests if the project is set correctly to the user

**2.1.8 public void ValidateEmailTest()**

Test if the email is validated correctly

**2.1.9 public void ValidateCNPTest()**

Test if the cnp is validated correctly

**2.2.1 public void MarkTest()**

Test is the mark is correctly set