| Assignment 1 | | Project Summary | |
| --- | --- | --- | --- |
| Course | | **Web Application & Service Development with Spring Framework v5** **- 2020** | |
|  | | | |
| Project author | | | |
| № | Pseudonym | | Face-to-face/ online |
| 1 | Viktor Aleksandrov | | online |

| Project name | Education Management System (EMS) |
| --- | --- |

| 1. Short project description (Business needs and system features) |
| --- |
| Piles of diaries, notebooks, countless written pens, and dozens of hours spent checking homework and exams are an invariable part of a teacher's job nowadays. Lost homework notebooks, missed deadlines, long waits for diplomas and certificates - this is the reality for all students these days. The troubles for parents are no less than the above. Educational processes in small and large schools in our country are quite outdated and time-consuming. For this purpose, an **Education Management System*(EMS)*** will be developed. This system aims to make life and work easier for teachers, students, and parents.  The system will be developed using ***Spring 5 Application Development Framework***.The backend will be implemented as a ***REST/JSON API*** using JSON data serialization. There will also be real-time event streaming from the server to the web client using ***WebFlux*** and ***Server Sent Events (SSE)/WebSocket***. The main user roles (actors in UML) are:  • *Headmaster* (extends *User*) – has the ability to manage all the school resources.  • *Teacher* (extends *User*) – can create exams and homework for the students in the specified school year. The teacher has the ability to grade the students by their homework as well. The *Teacher* can only view their Timetable  • *Student* (extends *User*) – can submit homework, take exams, and view all the details for their timetable, grades, and subjects  • *Parent* (extends *User*) – can view information for timetable, grades, subjects, upcoming exams, and pending homework for their child |

| 1. Main Use Cases / Scenarios | | |
| --- | --- | --- |
| **Use case name** | **Brief Descriptions** | **Actors Involved** |
| * 1. **Browse information** | *Headmaster* can browse all the available information within the EMS.  *Teachers* can browse information on homework, upcoming and active exams only created by them, upcoming competitions. They can browse information for students as well.  *Students* can browse information for their grades, certificates, upcoming exams and competitions, pending homework, and timetables.  *Parents can browse all available information (the same that the student can see) that concern their children only.* | All users |
| * 1. **Update profile information** | *All Users* can view and edit their personal *User Data* except the email address, id, and role | All users |
| * 1. **Edit user’s profile information** | *Headmasters* can edit all the user’s profile information except the email address, id, and role | *Headmaster* |
| * 1. **AddUser** | *Headmasters* can add new *Users* to the EMS. | *Headmaster* |
| * 1. **Manage school year** | *Headmasters* can manage all the school years that are already defined within the EMS (1 - 12), which includes assigning teachers, editing timetables, and managing subjects, and students. | *Headmaster* |
| * 1. **Create exams** | *Teachers* can create new exams and assign them to a specified school year | *Teacher* |
| * 1. **Create homework** | *Teachers* can create new homework and assign them to a specified school year | *Teacher* |
| * 1. **Grade student** | *Teachers* can grade students by their homework and exam results or any other criteria. | *Teacher* |
| * 1. **Take exam** | *Students* can take exams | *Student* |
| * 1. **Submit homework** | *Students* can submit and re-submit assigned homework | *Student* |
| **2.11. Announce Competition** | *Headmasters* can announce(create) new competiotions to the *Students* | *Headmaster* |
| **2.12. Participate Competition** | *Students* can participate in upcoming competitions | *Student* |
| **2.13. Receive Certificate** | *Students* can receive *Certificates* for winning in a competition | *Student* |
| **2.14. Assign Certificate** | *Headmasters* can assign *Certificates* to *Students* | *Headmaster* |

| 1. API Resources (Backend) | | |
| --- | --- | --- |
| **View name** | **Brief Descriptions** | **URI** |
| * 1. **Users** | GET *User Data* for all users. Available only for *Headmasters and Teachers*.  POST new *User Data* (Id is auto-filled by *EMS* and modified entity is returned as result from POST request). Available only for *Headmasters*. | */api/users* |
| * 1. **User** | GET, PUT, DELETE *User Data* for *User* with specified *id*, according to restrictions described in UCs. | */api/users/{id}* |
| * 1. **Login** | POST *User Credentials* (id and password) and receive a valid *Security Token* to use in subsequent API requests. | */api/login* |
| * 1. **Logout** | POST a logout request for ending the active session with *EMS,* and invalidating the issued *Security Token*. | */api/logout* |
| * 1. **Students** | GET All *Students* within EMS, according to restrictions described in UCs. (*The POST request can be done only with specified school year index - see 3.6.)* | */api/students* |
| * 1. **Students By School Year** | GET assigned *Students* within this school year with specified *schoolYearIndex*, according to restrictions described in UCs and POST new *Student*(Id is auto-filled by *EMS* and the modified entity is returned as the result of the POST request). | */api/schoolYears/{schoolYearIndex}/students* |
| **3.7. Student** | GET, PUT, DELETE *Student* Data for *Student* with specified *id* and return the managed *Student. (An alternative to this is 3.3, but I preffer having URI limiting the users to Students only.)* | */api/students/{id}* |
| **3.8. Teachers** | GET All *Teachers* within EMS according to *User's Role* and identity security restrictions and POST new *Teacher*(Id is auto-filled by *EMS* and the modified entity is returned as the result of the POST request). | */api/teachers* |
| **3.9. Teachers By Subject Name** | GET *Teachers* with specified *subject name* according to *User's Role* and identity security restrictions. | */api/teachers/{subjectName}* |
| **3.10. Teacher** | GET, PUT, DELETE *Teacher* with specified *id* according to *User's Role* and identity security restrictions. | */api/teachers/{id}* |
| **3.11. School Years** | GET *School Years Data* within the EMS according to *User's Role* and identity security restrictions. There won't be any POST methods here, the school years are pre-defined | */api/schoolYears* |
| **3.12. School Year** | GET, PUT, DELETE *School Year Data with the specified schoolYearIndex* within the EMS, according to *User's Role* and identity security restrictions. | */api/schoolYears/{schoolYearIndex}* |
| **3.13. Exams** | GET exams(according to *User's Role* and identity) and POST new *Exams*(Id is auto-filled by *EMS* and the modified entity is returned as result from POST request). | */api/exams* |
| **3.14. Exams By School Year** | GET *Exams*(according to *User's Role* and identity) and POST new *Exams*(Id is auto-filled by *EMS* and the modified entity is returned as the result of the POST request). | */api/schoolYears/{schoolYearIndex}/exams* |
| **3.15. Exam** | GET, PUT, DELETE *Exam Data* for *Exam* with specified *id* and return the managed *Exam.* | */api/exams/{id}* |
| **3.16. Exam Results** | GET *Exam Results* (according to *User's Role* and identity) ) for *Exam* with specified *id.* | */api/exams/{id}/results* |
| **3.17. Homeworks** | GET All *Homework Data* by specified *schoolYearIndex,* POST new *Homework*(Id is auto-filled by *EMS* and the modified entity is returned as result from POST request). | */api/homeworks* |
| **3.18. Homeworks By School Year** | GET All *Homework Data* by specified *schoolYearIndex,* POST new *Homework*(Id is auto-filled by *EMS* and the modified entity is returned as result from POST request). | */api/schoolYears/{schoolYearIndex}/homeworks* |
| **3.19. Homework** | GET, PUT, DELETE *Homework* by specified id and return the managed *Homework.* | */api/homeworks/{id}* |
| **3.20. Competitions** | GET All *Competitions Data,* POST new *Competition*(Id is auto-filled by *EMS* and the modified entity is returned as result from POST request)*.* | */api/competitions* |
| **3.21. Competition** | GET, PUT, DELETE *Competition* by specified *id* and return the managed *Competition.* | */api/competitions/{id}* |
| **3.22. Certificates** | GET All *Certificates,* POST new *Certificate*(Id is auto-filled by *EMS* and the modified entity is returned as result from POST request)*.* | */api/certificates* |
| **3.23. Student Certificates** | GET All *Certificates* that are won by the *Student* with the specified *id,* POST new *Certificate*(Id is auto-filled by *EMS* and the modified entity is returned as result from POST request)*.* | */api/students/{id}/certificates* |
| **3.24. Certificate** | GET, PUT, DELETE *Certificate Data* for *Certificate* with specified *id* and return the managed *Certificate .* | */api/certificates/{id}* |