NAME : CourseSeeker.io

Total estimation: 18.2 story points (145.6 hours of work)

Team members: Alessandro, Luca, Daniil, Safiyye, Dilay

REQUIREMENTS

DESCRIPTION:

The system is a web application serving as an education platform.
The core functionality is the following: teachers can create courses, a course is a board where various resources can be published (text, files, videos); students can view courses, get the resources, send messages on a chat. Our goal is to reach security by authentication and to have a minimalistic web design approach.

REFERENCE WEBSITES:

- Moodle
- MIT OpenCourseware

User story structure: *"As a* [type of user], *I want* [functionality], *so that* [benefit] ...."

---

1. *As a* **user**, *I want* **to authenticate myself**, *so that* **I can access the web application pages and do stuff my role permits me to do (student, teacher)**.

- Users can log in with their credentials.
- Unauthorized users should be redirected to the login page when trying to access restricted content.
- Logout functionality should be available.
- Error messages should be displayed for incorrect login attempts or unauthorized access.
- Optional: Support for password reset and account recovery via email.

Use case (user):
IF a user is not in logged in state

- A login form made of user and password fields is displayed
- User enters credentials
- User clicks login button

- IF the credentials are wrong, an error message is shown; ELSE user is enters logged in state

IF a user is in logged in state

- User is redirected to homepage

Story points: 1.6

---

2. *As a* **user**, *I want* **to browse courses**, *so that* **I can access learning materials**.

- Users can see a list of available courses.
- Users can enroll in or leave courses.
- Only enrolled students can access course materials.
- Users can filter for just enrolled courses
- Optional: require simple password pre-shared by teacher by other means to enroll

Use case (user):

IF user is on available courses page

- A list of courses cards is displayed (each containing title, teacher name, short description) (If just enrolled courses filter is enabled, just enrolled courses are displayed)
- User can search them
- Users can click on one course card to be directed to the corresponding course page.
- A filter button permits to show just enrolled courses instead of all courses.

Story points: 3

---

3. *As a* **teacher**, *I want* **to create a course**, *so that* **I can have courses**.

- Teachers have a button to create new courses.
- The creation process asks for information like title, description.

Use case (teacher):

IF a **teacher** is on the profile page

- Course create button available in the menu.
- As you click the button:
- A form is displayed asking for title, description, contents, credits, price etc.
- A create button at the bottom to submit the creation.

Story points: 2

---

4. *As a* **user**, *I want* **to view a course page**, *so that* **I can access the material published**.

- A course page has a header section containing the title, a description, and messages left by the teacher.
- A course page has a chat component.
- A course page has a list of material resources (files).
- A button permits to quickly share the course page via a link.
- If a user is not enrolled the material is hidden (not sent by the server) and a panel with a button to enroll (with potential password required) will be displayed.

Use case (user):
IF is not enrolled:

- Visualize the general description of the course and the content.
- Be able to share the link
- Be able to enroll the course (maybe with password)

IF is enrolled:

- Visualize and access the course resources and the date of the uploading.
- Visualize the announcements from the teacher.

Story points: 3

---

5. *As a* **student**, *I want* **to connect to a resource page**, *so that* **I can see and download it**.

- View the resources uploaded by the teacher

Use case (student):

IF student is on a resource page

- The resource title and description is displayed.
- Download button in case of files/documents/videos.

Story points: 1.6

---

6. *As a* **teacher**, *I want* **to be able to modify my courses pages**, *so that* **I can publish my material**.

- Teachers browsing a course page have editing/uploading functionalities available.

Use case (teacher):

IF teacher is on one of its course page

- A button is shown after the last resource to create a new material
- When a teacher clicks it and is shown a form to insert title, description and upload the file.
- Only the teacher can delete it

Story points: 1.4

---

7. **As a teacher**, I want to be able to collect files from students (for assignments), so that I can assign a mark to it.

- Teachers can create an assignment where students can upload a file as a completed task.
- Teachers can set a deadline for assignment completion.

Use case (teacher):

IF teacher is one of its course page

- A button is shown after the last resource to create a new assignment
- Teacher clicks it and is shown a form to insert title, description and deadline.

Use case (student):

IF a student is on a course page

- Student can click on the assignment card
- An upload form is displayed to make him upload its file

Story points: 1.6

---

8. **As a user, I want to communicate with other users,** so that I can ask questions and discuss course-related topics.

- Users can send messages on the course chat.

Use case (user):

IF a user is on a course page

- At the top of the page a chat is available
- Users can send messages and read other users messages.

Story points: 4

NON-FUNCTIONAL REQUIREMENTS:

1. Security

- Authentication should use secure password hashing and storage.
- Session management should be implemented (e.g., token-based authentication or session cookies).

2. Performance

- Pages should load fast
- Buttons and components should be responsive

3. Compatibility

- Pages should work well on different browsers
- Pages should work well on different devices with different screen sizes

4. Reliability

- The application backend should not crash
- In case of errors it should log informations and continue working as well as it can

5. Scalability

- Many users should be able to use the system concurrently without affecting user experience