ZOOL412 Project

Technical Specifications for KPI Company website and Staff Portal

Nathan Hollows

1 Project Overview

Objective: Build a website for ZOOL412 that serves as both a fictional corporate website and an interactive portal for students to receive assignments, submit reports, and receive feedback. The website will increase the level of simulation by presenting a more realistic and corporate environment for students to engage with. It will achieve this by providing a corporate front-facing website with KPI's mission statement and staff profiles, as well as an "Employee Portal" for students to submit their work, receive feedback, and communicate with fictional employees in the company.

Stakeholders:

- ZOOL412 Lecturers
- ZOOL412 Students

2 Functional Requirements

2.1 Front-End

2.1.1 Corporate Front-end

The following pages are the minimum requirements for the corporate front-end:

- Home Page with mission statement and thematic content.
- Employee Portal Login Page

Navigation: Home, About, Portal Login.

2.1.2 Employee Portal

The following pages are the minimum requirements for the employee portal:

- Dashboard
  - Current assignment.
  - Notices indicator.
  - A link to the user's current role description.
- Assignments
  - List of assignments.
  - Assignment introduction.
  - Executive brief download.
  - Submit "Expert Opinion" report.
  - Submit "Research Funding" request.
  - Submit report (file upload).
- Directory: staff profiles.

- Data drops: raw data for assignments.
- Founders: company history.
  - An in-fiction way to incorporate the names of everyone who has worked on the project.

In addition to the above features, the portal could include:

- TeamLink: team chat.
- Notices: read-only notices from the company.
- Budget
  - Currently included in the brief download but could be improved.
- Stocks
  - Currently included in the brief download but could be improved.
- Role-based access control, e.g., only team-leads can submit reports.

2.1.3 Admin (Lecturer) Portal

- Create, manage, and delete assignments.
- View and provide feedback on submitted reports.
- Manage student accounts.
- Manage student teams.
- Assign a team lead for each team.
- View and respond to student messages via the internal mail system.
  - This may be in the style of a forum, with threads for each team, to decrease cognitive load for lecturers.

Future iterations may include:

- Website content management system.

2.2 Back-End

2.2.1 User Authentication

- Login via Microsoft 365 OAuth.
- Role differentiation (Student, Lecturer).

2.3 User Stories

2.3.1 Corporate Front-End

1. **Home Page**
   - As a **visitor**, I want to view the company's mission statement and thematic content so that I can understand the purpose and scope of KPI.
2. **About Page**
   - As a **visitor**, I want to learn about KPI's history and background information so that I can immerse myself in its fictional context.
3. **Staff Page**
   - As a **logged-in employee**, I want to view staff profiles, photos, and bios so that I can learn about the fictional team members.
   - As a **logged-in employee**, I want the ability to click a link in a staff profile to initiate a chat with the fictional staff member.

2.3.2 Employee Portal

The following user stories follow the fictional context of the project, where students are employees of KPI:

1. **Dashboard**
   - As an **employee**, I want to see my current assignment so that I know what I need to work on.
   - As an **employee**, I want to view my submission status so that I can confirm whether I've successfully submitted my work.
   - As an **employee**, I want to see feedback on my submissions so that I can improve my work.
   - As an **employee**, I want a notification icon for new messages in my mailbox so that I can stay updated with communication.
2. **Report Submission Form**
   - As the **team lead**, I want to submit a report on behalf of my team with a file upload feature so that I can complete assignments.
   - As a **team-member**, I want to be notified if I am not authorised to submit reports so that I understand my role.
3. **Feedback Section**
   - As an **employee**, I want to view feedback provided by my manager on specific assignments so that I can learn and improve.
4. **Internal Message System**
   - As an **employee**, I want to send messages to employees so that I can ask questions about my assignments.
   - As an **employee**, I want to receive messages from staff members so that I can stay informed.
5. **Role-Based Access Control**
   - As a **team lead**, I want exclusive access to submit reports so that only authorised individuals handle team submissions.
   - As a **student**, I want to view and contribute to team discussions without overriding the role of the team lead.

2.3.3 Manager (Lecturer) Portal

1. **Assignment Management**
   - As a **lecturer**, I want to create, edit, and delete assignments so that students have up-to-date tasks.
   - As a **lecturer**, I want to set deadlines for assignments so that students understand the timeline for submissions.
2. **Feedback on Reports**
   - As a **lecturer**, I want to view submitted reports so that I can evaluate the students' work.
   - As a **lecturer**, I want to provide written feedback on reports so that students know how to improve.
   - As a **lecturer**, I want to upload additional feedback files for detailed comments or examples.
3. **Student Account Management**
   - As a **lecturer**, I want to create and manage student accounts so that each participant has access to the portal.
   - As a **lecturer**, I want to assign students to teams so that they can collaborate on assignments.
   - As a **lecturer**, I want to assign team leads to ensure accountability within each group.
4. **Internal Message System**
   - As a **lecturer**, I want to view and respond to student messages via an internal mail system so that communication remains centralised.
   - As a **lecturer**, I want team threads to be organised in a forum-like style to manage

conversations efficiently.

2.3.4 Back-End

1. **User Authentication**
   - As a **student**, I want to log in with my Microsoft 365 credentials so that I can securely access the portal.
   - As a **lecturer**, I want to log in with my Microsoft 365 credentials so that I can access administrative tools.
2. **Role Differentiation**
   - As a **student**, I want to access student-specific pages and tools so that I only see content relevant to me.
   - As a **lecturer**, I want access to admin features like assignment creation and feedback tools so that I can manage the course effectively.

3 Database schema

The following is a proposed database schema for the project. This schema is subject to change based on the developer's discretion. This was written in DBML format and can be visualised in dbdiagram.io.

Note: This is not entirely up-to-date with the current project requirements.

```
Table users {
    id int [pk, increment]
    name varchar
    email varchar [unique]
    role enum('student', 'lecturer')
    team_id int [ref: > teams.id, null]
    created_at datetime
}

Table teams {
    id int [pk, increment]
    name varchar
    lead_user_id int [ref: > users.id] // Manually assigned team lead
    created_at datetime
}

Table assignments {
    id int [pk, increment]
    title varchar
    description text
    created_at datetime
    due_date datetime
    file_path varchar
}

Table reports {
    id int [pk, increment]
    submitted_by int [ref: > users.id]
    team_id int [ref: > teams.id]
    assignment_id int [ref: > assignments.id]
    file_path varchar
    submitted_at datetime
    feedback text [null]
    feedback_at datetime [null]
    feedback_file_path varchar [null]
}

Table staff { // Table for fictional employees
    id int [pk, increment]
    name varchar
    role varchar
    bio text
    created_at datetime
}

Table messages {
    id int [pk, increment]
    student_id int [ref: > users.id]
    staff_id int [ref: > staff.id]
    is_from_student bool [default: true] // Flag to indicate if the message is from a live
    subject varchar
    body text
    created_at datetime
    read_at datetime [default: null]
}

Table user_oauth {
    id int [pk, increment]
    user_id int [ref: > users.id]
    provider varchar
    provider_id varchar
    token text
    created_at datetime
}
```

4 Non-Functional Requirements

- **Security**:
  - Use HTTPS for all connections.
  - All data **should** be stored on University servers.
- **Accessibility**:
  - Follow WCAG 2.1 AA standards for accessibility.
  - The colour scheme is currently AA, save for the middle of the "screen" gradient where it is A.
- **Browser Compatibility**:
  - Chrome, Firefox, Safari, Edge (latest versions).
  -
    * Mobile.

5 Implementation Plan

5.1 Technology Stack

Left to the discretion of the new developer.

5.2 Milestones

1. **Wireframing**:
   - Create mockups for the front page, portal, and admin sections.
2. **Back-End Development**
3. **Testing**:
   - Unit tests for all components.
   - End-to-end testing of user flows.
4. **Deployment**:
   - Deploy to University servers for testing the live environment.

6 Testing Strategy

- **Unit Testing**:
  - Test individual components and functions.
- **Integration Testing**:
  - Validate communication between front-end and back-end.
- **User Acceptance Testing**:
  - Ensure usability by having test users (e.g., lecturer and previous students) navigate the system.
- **Performance Testing**:
  - If necessary, however, user load is expected to be low.