Finalized D1

CS 319

2024-25 Fall semester

S3T4 - Code Busters

Umay Dündar 22202573

Elif Ercan 22201601

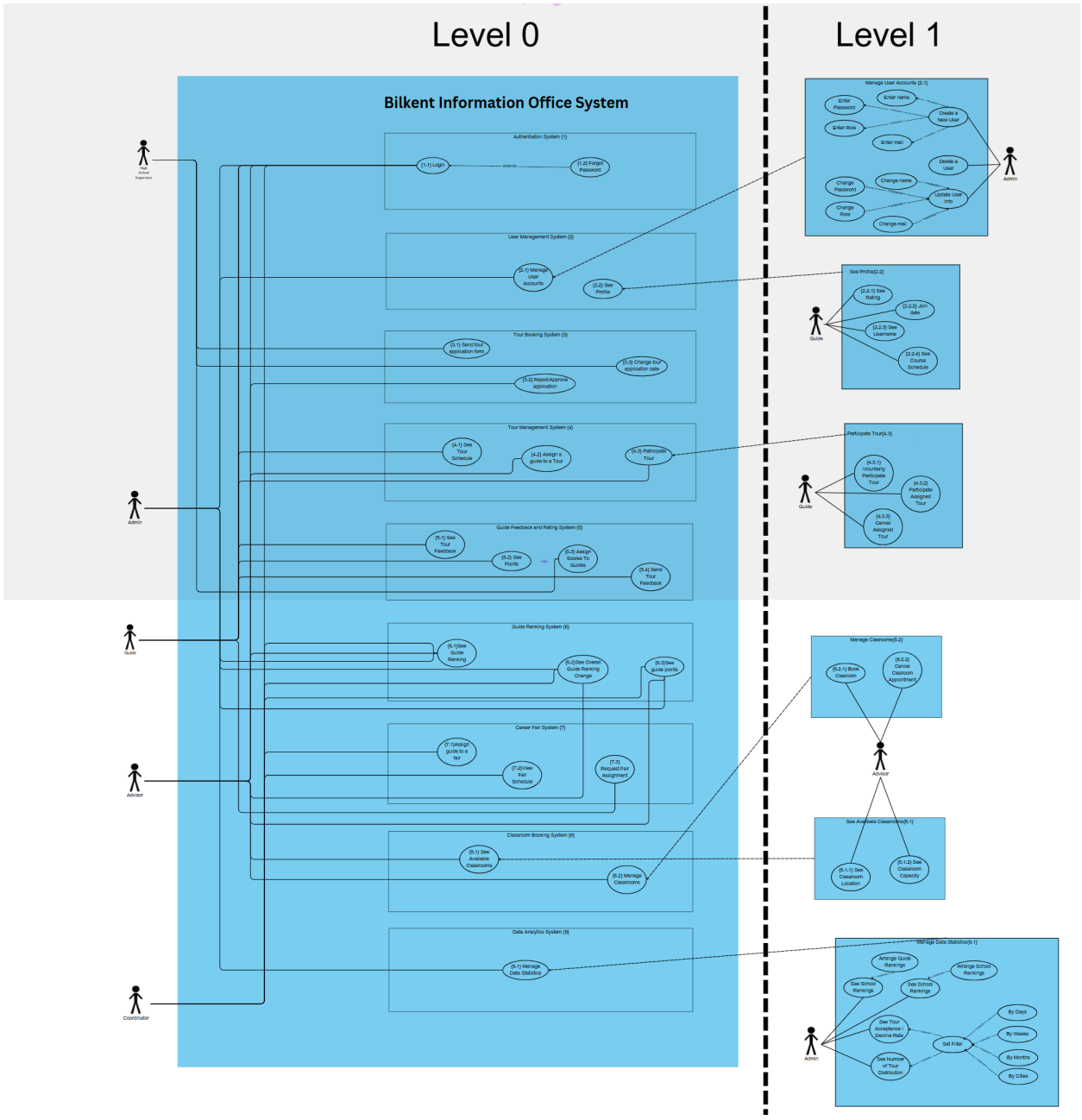
İbrahim Barkın Çınar 22003874

Kemal Onur Özkan 22201820

Contents

1 Use Cases

1.1 Use Case UML Diagram (Level 0 + Level 1)



1.2 Use Case Stories

1.2.1 Admin:

As an admin, I want to set who the users are and their roles so that everyone has the right responsibilities.

As an admin, I want to create and delete users of every type so that I can add new guides, advisors, and coordinators to the system and remove the old guides, advisors, and coordinators from the system.

As an admin, I want to see all the guides' feedback and overall scores for each tour so that I can determine the issues and track the guides' performances.

As an admin, I want to see the guide's rankings so that I can track their overall performance with respect to each other.

As an admin, I want to manage data statistics about users of the system, tours and schools so that I can make informed decisions about the tour system.

1.2.2 Advisor:

As an advisor, I want to see all guides' schedules to assign the available guides to the tours that don't have a voluntary guide.

As an advisor, I want to book classrooms and send groups to these classrooms.

As an advisor, I want to see the available classrooms with their location and capacity to program the tour routes and QA sessions accordingly.

As an advisor, I want to be able to see all the guides' feedback and overall scores for each tour so that I can determine the issues and track the guides' performances.

As an advisor, I want to see the guides' rankings to track their overall performance with each other and assign guides to the tours accordingly.

1.2.3 Coordinator:

As a coordinator, I want to see the guides' feedback so I can decide which guide to assign to each school's university fair.

As a coordinator, I want to determine who attends fairs so that the most qualified and available person attends each fair.

As a coordinator, I want to see guides' rankings to assess them accordingly.

As a coordinator, I want to create and delete guide and advisor users to add new guides and advisors to the system and remove the old guides and advisors from the system.

As a coordinator, I want to be able to assign guides to career fairs so that they can represent Bilkent there.

1.2.4 Guide:

As a guide, I want to select available hours in a week according to my course schedule to give a tour when I am available.

As a guide, I want to see my schedule so that I can plan my week based on my schedule.

As a guide, I want to voluntarily select tours through the system so that I can participate in the tour.

As a guide, I want to see the feedback given by high schools after each tour so that I can improve my skills with the feedback.

As a guide, I want to see my ranking in the other guides so that I can improve my performance

As a guide, I want the system to automatically update my scorecard so that I can track how many tours I give until the end of the month.

1.2.5 High School Supervisor:

As a high school supervisor, I want to book tours so my students can visit the campus and receive a tour.

As a high school supervisor, I want to be able to cancel a booked tour so that students can visit the campus at another possible time.

As a high school supervisor, I want to be able to change the date and time of the booked tour so that students can visit the campus at another available time.

1.2.6 High School Student:

As a high school student, I want to book a tour to visit Bilkent individually or with a small group.

As a high school student, I want to cancel a booked tour so that I can visit the campus at another time that is convenient for me.

As a high school student, I want to change the date and time of my booked tours to visit Bilkent at another time possible.

2. Non-Functional Requirements

2.1 Security

- Users other than the schools must be authenticated with a username and password combination.
- Passwords will be saved in hashed form in the database.
- Only the admin and coordinator users can create and delete other user accounts, ensuring that outside users are not in the management system.
- Every user in the system has a specialized role, and only the related part of the website is shown to them, which also maintains security.

2.2 Performance

- The website should respond in up to 1 second after a click.
- The web app should load within 3 seconds on standard broadband.
- Search results within the app should appear within 2 seconds.
- Daily, weekly and monthly tour schedules should be retrieved and loaded in less than 1 second.

2.3 Usability

- The website should have a UI with colour coding that is concise and intuitive to use, which gives the page a high learning curve.
- The web app should provide helpful error and info messages to the users.
- Users can access their needs in less than 5 clicks on the webpage.

2.4 Scalability

- The website should be able to host data of over 100 users without losing performance.
- The software should support real-time data passing and retrieving between users who have different profile types, even if there are over 100 users.

2.5 Reliability

- The system should be available at least within the working hours.
- The system should be backed up regularly (like on a weekly basis or monthly basis) to be able to save critical information in a reasonable time (so that the system can be available during working hours).

2.6 Compatibility

- The system should be compatible with the common web browsers.
- The system should work on different computers.

2.7 Accessibility

- The website should be accessible and easy to use, which ensures users with different technical capabilities can easily interact with it.
- All interactive elements should be of a reasonable size, and form inputs should have informative labels that increase the accessibility of the website for outside users.

2.8. Maintainability

- The web app should have clean code with sufficient comments to explain the code, making the future maintainers' jobs easier.
- The code should be modular and well-organized, which will make future code changes, bug fixes and feature additions faster.

2.9 Data Integrity

- Ensure that database data is accurate, consistent, and updated in real-time.
- Data should be backed up in case of emergency.
- Partial manipulation of the database instances is not allowed in most cases to ensure that no data change happens when a data loss occurs.

3. Tech Stack

We will use the **MERN (Mongo, Express, React, Node)** tech stack to create this project. We chose this tech stack because it is commonly used and has many usable libraries and interfaces that can be used in our project.

Frontend: React.js

API: Express

Backend: Node.js

Database: MongoDB