Bilkent University
Department of Computer Engineering
CS-319 Deliverable 3
19 December 2024

Fall 2024-2025

Rida Fatima
Emine Fidan
Mert Özkaya
Fazlı Güdül
Ömer Yaslıtaş

Design Goals and their Trade-offs:

1. Security and Data Privacy
    a. Justification
        i. The system caters to various different users, including coordinators, advisors, guides, trainees, and applicants (high school and individual). This leads to the handling of sensitive data, including high school counselor contacts, guide schedules, trainee progress, and visitor feedback. Ensuring reliable data security is critical to maintain user trust and compliance.
    b. Benefits
        i. Trust: By ensuring that only authorised personnel have access to user data, user trust in the system is maintained.
        ii. Compliance: Reliable security systems meet protection laws and avoid financial and legal penalties.
        iii. Integrity: Reduces risk of unauthorised data access
    c. Implementation
        i. Encrypt sensitive data (for instance, passwords and user data) in the database
        ii. Implement role-based access to ensure that users only have access to the data they have authorization for. For instance, trainees should not have the same data access that the coordinator has.
    d. Trade-offs
        i. Impact on performance: Security mechanisms, including encryption, decryption and authorisation are time-consuming and might decrease
        system performance. Real-time encryption for data (for instance, during form submissions or database queries) adds latency and decreases efficiency. Frequent checks for user identity or session validation can slow down operations.
        ii. Complexity during development: Ensuring that system security is reliable includes thorough testing during development, including vulnerability scans to ensure that there are no weak-points in the security mechanisms. Testing for edge-cases must be done which further adds complexity during the development phase.

iii. User Experience: Security systems, especially those that require user involvement, such as two-factor authentication or CAPTCHA, can make the user experience less satisfactory as it decreases efficiency. Strict password policies, such as the necessary inclusion of special characters, and both uppercase and lowercase letters, can make the experience frustrating for users. Session timeouts may interrupt workflows and decrease user satisfaction.

2. Reusability
    a. Justification
        i. Our system is designed to accommodate multiple users (coordinators, advisors, guides, trainees, and applicants) and various functionalities, including analyzing user feedback, organizing data, and user authentication. To lessen the effort it takes during the development process and to ensure scalability, the reusability of code, components and designs is prioritized. By utilizing reusable models (e.g. user authentication, data handling, role-based access), similar functionalities can be implemented efficiently across the website without having to rewrite code every time.
    b. Benefits
        i. Faster Development: Using reusable components significantly reduces the time it takes for development since the developers do not need to build functionalities from scratch. For instance, a single role-based access control module can be reused for different user types.
        ii. Maintainability: Reusable components make the system easier to maintain since updates and bug fixes only need to be applied to one part of the system. For instance, updating the authentication logic affects all parts of the system that use it.
        iii. Consistency: Reusable modules have consistent functionality and design throughout the system which improves reliability of the system. Predictable system behavior also improves user experience.
        iv. Scalability: Reusable components make it easy to expand the system when there are new requirements, such as integrating new features or accommodating additional user roles.

c. Implementation

i. For the front-end development of the website, there were certain components that were the same across all pages for each user. For instance, the sidebar for coordinator dashboard is the same for all coordinator pages. So, instead of implementing the sidebar in each page separately, we have an independant sidebar class that all the pages import in their classes. This way, when there is a bug to fix in the sidebar, it’s easy to just fix it in one sidebar class compared to if there was a sidebar class separately for each page. This reduces the time it takes to fix bugs and improve the UI.

d. Trade-offs

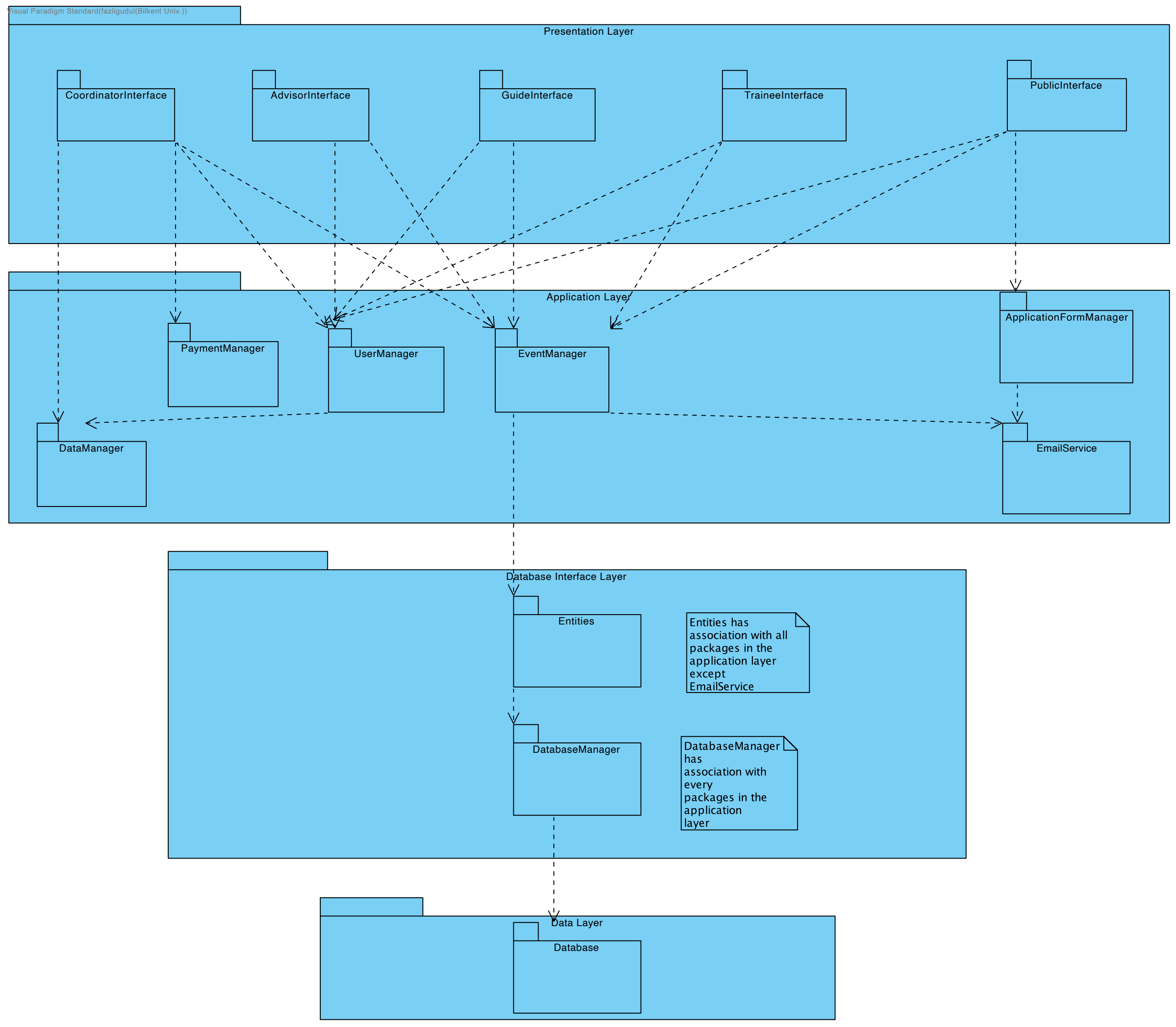
i. Increased Initial Development Effort: Designing for reusability requires careful planning. This increases the complexity and time required during the initial development phase.

ii. Performance Overhead: Reusable components may not be as optimized for specific use cases as custom-built ones for each case. For example, a single data validation module used for multiple forms may have checks not required for a particular form, leading to reduced efficiency.

iii. Complexity in Maintenance: Changes to a reusable component might consequently affect multiple parts of the system where it is used. This introduces a risk of unintended side effects.

iv. Learning Curve for Developers: Developers must understand the reusable components, their interfaces, and constraints. For new team members, the learning curve to use these components effectively can be steep.

Subsytem Decomposition Diagram



Subsystem Explanations

ApplicationFormManager

This subsystem is responsible for managing application forms submitted by counselors/students.

- Keeps track of active application forms.
- Assigns reviewers to forms and updates the status of the forms.
- Sends confirmation emails to applicants and archives completed forms.

EventManager

Handles the management of events, which may include fairs, group tours, or individual tours.

- Keeps records of active events and assigns guides to them.
- Sends requests to guides and updates the status of events as necessary.
- Archives completed events.

UserManager

Manages all user-related functionalities.

- Adds guides, advisors, or trainees to the system. Handles user authentication.
- Logs the latest activities of users and enables updates to user profiles.
- Sends notifications to users as required.

DataManager

Manages and retrieves data for reporting and analysis purposes.

- Updates high school information and maintains archived application forms, events, and tour participant surveys.
- Provides data analytics for further insights.

PaymentManager

Oversees payment processing in the system.

- Tracks pending payments and marks them as completed when processed.
- Maintains a record of past payments for review.