CS261 Coursework Design Document

Group 17: *Ben Lewis, Dan Risk, Edmund Goodman, Jay Re Ng, John-Loong Gao, Rahul Vanmali, Tomás Chapmann Fromm*

1 Introduction

Deutsche Bank requires a system prototype to support the mentoring process for employees. The principal purpose of this document is to broadly record the process documentation and design choices for said system prototype. For process documentation, this document will detail development methodologies in addition to workflow and project organisation. Design choices include a prototype user interface design and technical diagrams demonstrating user interactions between multiple users and the system. Furthermore, our testing strategy and technical descriptions such as choice of technologies used and system architecture will be covered in depth to illustrate more full idea of the final product.

2 Process documentation and planning

2.1 Development methodology

We opted for an agile development methodology based on the Scrum model. The development timeline will be broken down into a large initial planning phase and a series of week-long sprints. We will have multiple meetings each sprint cycle to assess progress (sprint review) and adapt the plan if required.

An agile methodology is optimal for a short timeframe as it allows rapid product development with concurrent implementation and testing so we can make effective use of the time available. It also allows for more flexibility if we face unexpected delays during development, which is likely with an inexperienced team. Producing a thorough plan before beginning the agile development process is suitable since deliverable deadlines are fixed, so ensuring work is completed on time is critical. Additionally, contact with the client is limited so requirements are unlikely to change and the project will not deviate much from the initial plan.

2.2 Project organisation and communication

We decided on a flat team hierarchy since we have similar levels of software development experience, self-managed but with well defined responsibilities. To communicate between meetings, we decided to use Discord, as it is already a popular choice within the team and is easy to learn for new users. It offers both text and voice chat as well as a number of features such as GitHub integration, so we are notified via Discord when on changes, for example when pull requests are made.

Discord can be used for online meetings however we aim to convene in-person at least twice a week, as this enables us to share ideas much more easily and maintain focus during meetings. At these meetings we can compare progress on the system with our plan, and assign tasks based on documentation or what tasks need doing in order to stay on schedule. We will use Trello to assist in the task management process, as it provides a visual representation of current project progress.

2.3 Project schedule

2.4 Risk management

Risk	**Impact**	**Likelihood**	**Severity**	**Mitigation**	**Contingency**	**Residual**
Team members unable to work	Smaller team until the member can resume work	2	7	Team members take precautions	Reduce scope and reallocate work across team	2
Task longer than expected	Project delayed until task completed	5	4	Allocate buffer time to allow overrunning tasks	Re-assign team members to load balance	4
Poor code quality	System fails testing or linting procedures	5	4	Code review and automated testing through CI/CD	Re-assign team memebrs to fix pair program	3
Requirements change	Design and existing code must be changed	3	8	Use an agile methodology to facilitate requirements change	Update project plan and proceed with new one	3

Code lost	Code must be re-written	1	7	Use ‘git‘ as version control, and remote backup to GitHub	Restore code from local or remote backups	1
Problems with dependencies	Component of project fails external library or technology	2	4	Choose technologies and dependencies carefully	Find replacement library or technology	2
Scope creep	Addition of unnecessary features causing growth of project scale	7	4	Include extra features in project timeline, and stick to it	Drop lowest priority features	5

2.5 System attributes

3 Technical description

3.1 Scope

Our system must satisfy a complex specification, so its development must be carefully planned and executed. However, the goal is to create a prototype, not a finished product. As such, various aspects which might be required for a system going into production are not needed within a prototype.

One example of this is limitations on the scalability, for example the number of concurrent users. It is much more important that key features are implemented than being able to support dynamically changing high numbers of concurrent users, as the prototype will likely only be tested by a few users at any time. However, this does not mean that the prototype should not be designed without a good basis for facilitating scalability later as the system moves through its lifecycle. Another example of this is the use of Deutsche Bank branding, which is explicitly not to be used, although it would be incorporated in the production system. Instead, a generic and simple theme should be used in the prototype, which could then be later replaced with the correct branding.

3.2 Technologies used

3.3 System architecture

3.3.1 Data model

3.3.2 Database design

3.3.3 API design

4 UI/UX design

4.1 Site design

4.1.1 Page design

4.1.2 Page heirarchy

4.1.3 UI design principles and accessibility

4.2 User-system interaction

4.2.1 Use case diagram

4.2.2 Sequence diagram

5 Testing

5.1 Test cases

5.2 Unit and integration testing

5.3 Acceptance testing

References