CheckinChildren Final Documentation

Table of Contents

Description

The goal of this project was to create a web application that can be used by daycare providers to keep track of children and communicate with parents/guardians in an efficient manner.

Process

Our team's process was very similar to XP learned in 427. We had weekly meetings as a group at a set time and would meet more when necessary such as before an iteration presentation presentation. At the weekly meeting before each iteration we would assign roles for this iteration as well as finalize the use cases and assign them to partners.

We decided to leave it up to individual partner to decide how they would complete the use cases assigned to them. We thought this was best because sometimes partners may have 1-2 big complicated use cases where working closely together would be best, but other times they may have many smaller unrelated use cases where dividing them up would be more efficient. All partners had to complete a thorough testing of their code as well for every use case. This included both HTTP and Unit Tests. The goal was to do test driven development but often times we were unable to do it especially with HTTP tests where the pages could be changing as we go and we would have to changes tests accordingly to match new designs.

Refactoring of the code was usually done at the end of iterations. For the most part we had set model of how to create new things that required little refactoring but when we met up before meetings and showed what and how we had done things we would occasionally find

things that could be refactored. Additionally when the iteration required refactoring we would all go through the code and find something that could be improved that we had previously made.

Overall we worked well together as a team and efficiently completed our goals with a less strict XP system that worked for us.

Requirements & Specifications

Use Cases Implemented:

- As a company, I can register for an account
- As a company, I can view and edit my own information
- As a company, I can create day care facility
- As a company, I can view my facilities.
- As a company, I can delete a facility.
- As a company, I can create managers for a day care facility
- As a company, I can view my managers.
- As a company, I can demote a manager to employee
- As a company, I can move a child to a different facility.
- As a company, I can view the logs for my facilities.
- As a company or manager, I can filter my logs by type.
- As a DCP, I can create profiles for children and link them to parent accounts.
- As a DCP, I can check children into our facility.
- As a DCP, I can check children out of our facility.
- As a DCP, I can view children's statuses
- As a DCP, I can alert guardians when their children are checked in by email.
- As a DCP, I can alert guardians when their children are checked in by text.
- As a DCP, I can create accounts for parents.
- As a DCP, I have an address and phone number and can edit it.
- As a manager, I can view my employee's information.
- As a manager, I can promote an employee to a manager.
- As a manager, I can view the logs of a facility.
- As a manager, I can view my employees.
- As a manager, I can create accounts for my employees.
- As a parent, I can view my own information

- As a parent, I can edit my own information
- As a parent, I can view my child's information
- As a parent, I can edit my child's information
- As a parent, I can change my password
- As a parent, I can view a week-old history of my child.
- As a parent, I can view who checked out my child.
- As a parent, I can notify the facility that I will be late to pick up my child.
- As a parent, I can check my child's status.
- As a parent, I can set my alert preferences.
- As a parent, I can add people who can also pick up my child.

Architecture & Design

Architecture UML

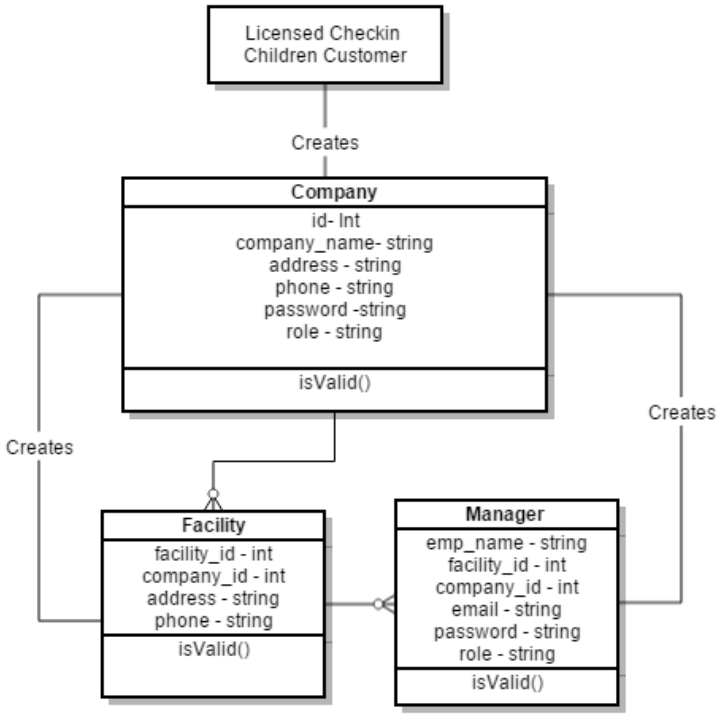

Figure 1: A new user creates a company account (in production this would be done by a CheckinChildren administrator after verifying credentials from an official request). A company creates and manages Facilities and Managers.

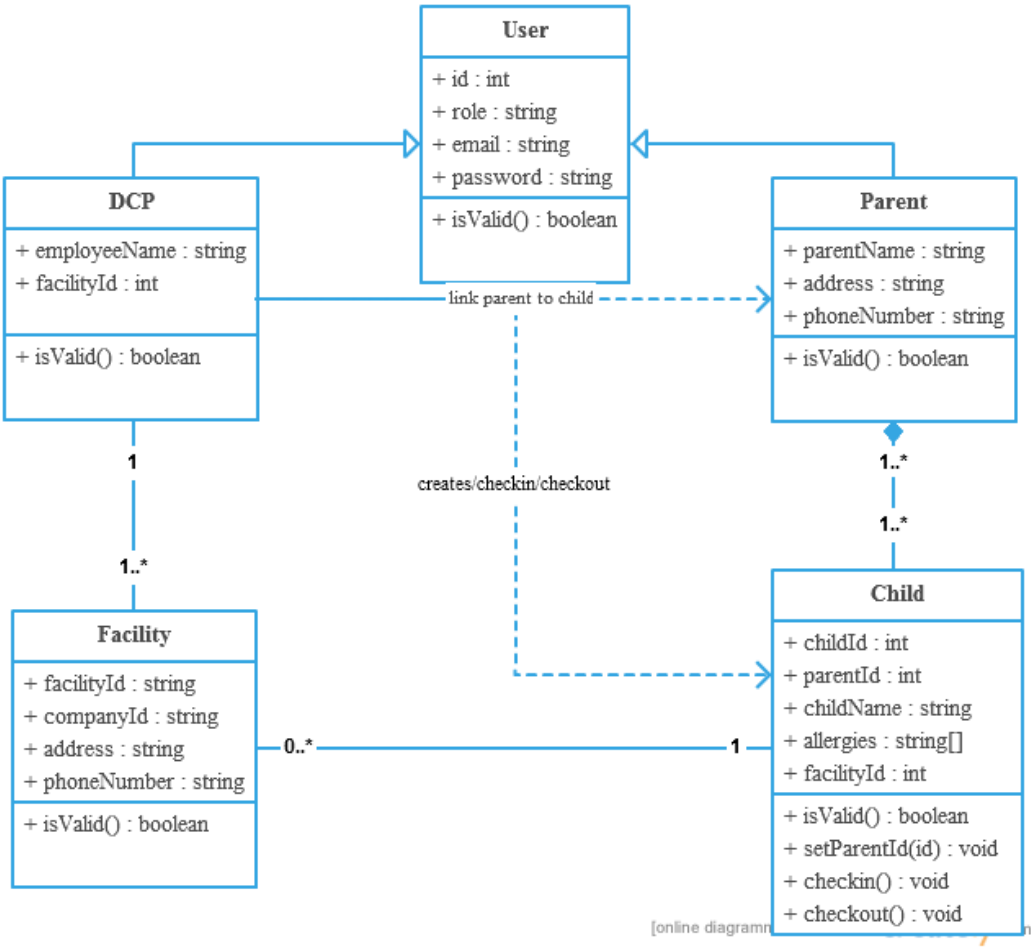

Figure 2: A Parent and DCP (Employee/Manager) are both Users with actions. DCP has an Associated Facility. Parent has associated Children. A DCP can perform checkin/checkout actions on these children associated with the same Facility.

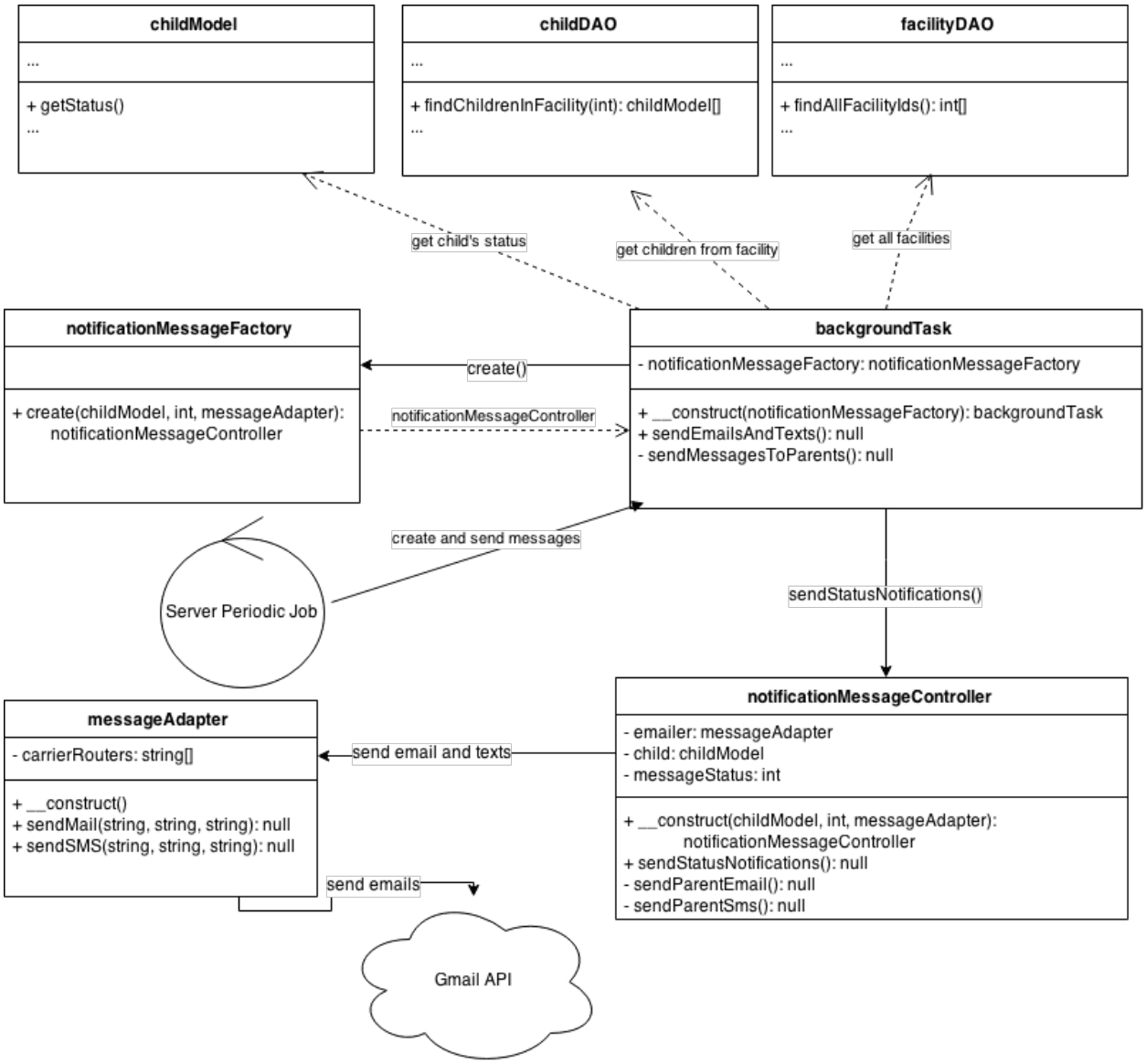

Figure 3: In production, the server runs a background task that periodically checks for child statuses and sends real-time updates to parents based on these statuses as necessary. The background task sends a message to the messageAdapter, which then sends a Gmail API query to send an email from our site's gmail account. Text messages are sent by sending emails to respective cell carrier routers (ie. phone_number@carrier_router.com).

High-Level Design

CheckinChildren is a web service and logistics system (somewhat similar to iTrust). It is implemented with a PHP/MySQL backend and standard HTML/CSS/Javascript on the frontend.

The general user flow of our system is demonstrated below. Our actors are a Company Account, Employee, Manager (which is an Employee with elevated privileges), and a Parent.

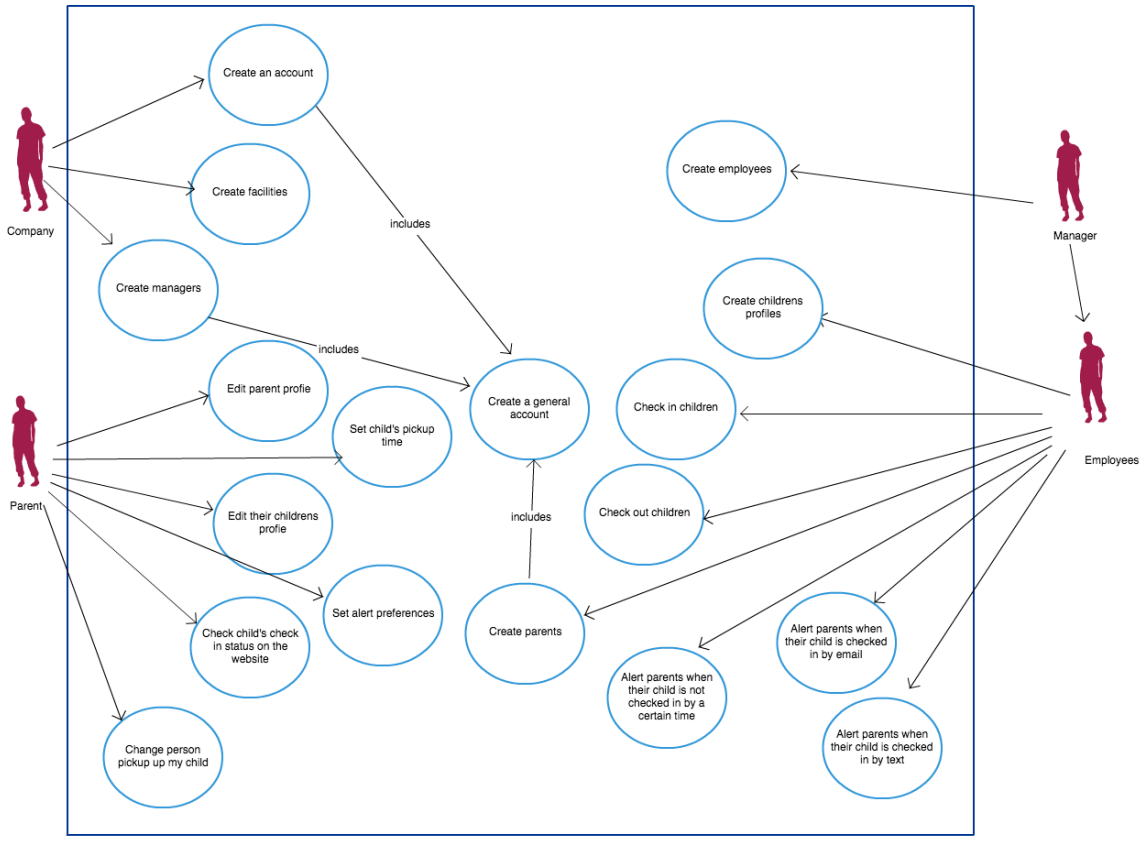

Figure 4: The general use is for day care companies to manage their facilities and employees. Employees in turn manage the children in their respective facilities, by checking them in and out of the system throughout the day as children arrive and leave. Parents can then check their children's current statuses, notify facilities of irregularities and late pickups, and receive periodic text/email updates when their children's statuses change (such as when the child is checked in and by whom, and when they are late to pick-up their children).

In addition, the system triggers notification messages, which give Parents dynamic updates to their child's current status.

MVC Code Pattern

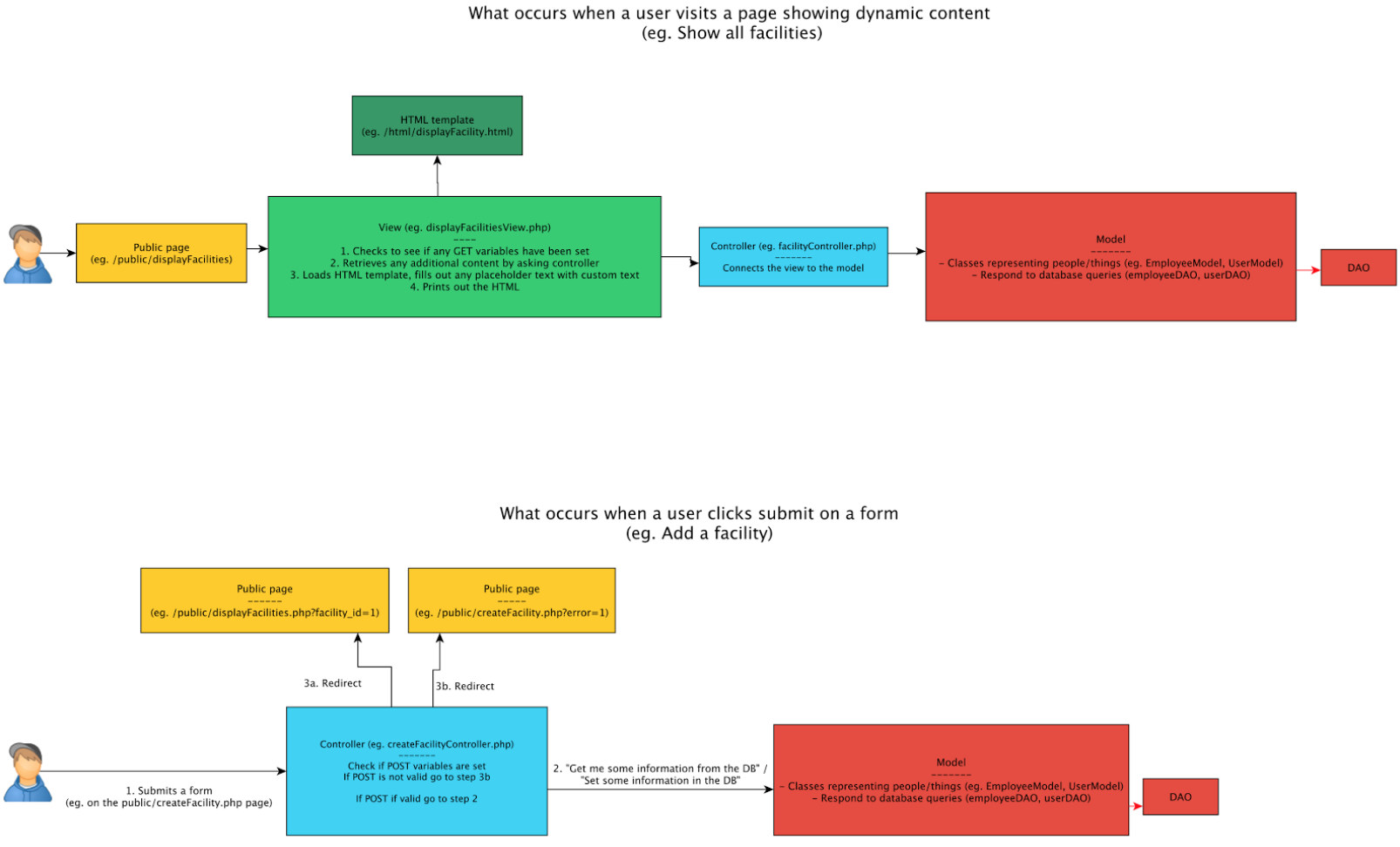

Figure 5: Our system adapted the Model View Controller architectural pattern.

Model Our model consists of SQL tables, Database Access Objects, and Object Models using the PHP language. They can be found under models in the scripts folder. At the deepest level, our information lies within SQL tables. In order to add to the database, user input is transformed into an object model by a controller. This is then added to the database by a DAO. To load from the database, SQL results are placed into objects to be observed or modified by the user presented by the view.

View The view is split into two parts and made up of PHP and HTML. Within the public folder, a page is constructed by loading various unique and shared parts to be viewed by the

user. For instance, everyone who logs in will see a header, a footer, and a sidebar. However, the actual content on the page varies by the role of the user and what page they are specifically accessing.

Within the view folder under scripts, the PHP that is loaded within the public files can be found. These files are named by the function they are displaying. They also may possess the functionality to pass information on to the controllers.

Controller Our controllers mainly consist of form handlers and can be found in the controllers folder under scripts. These files take information from the user obtained by the view, turn them into objects, and get them into our SQL tables using DAO functions.

We did not use a framework to implement our project. We did, however, use a framework to test our project. We used PHPUnit to unit test our model and DAO classes. PHPUnit allowed us to perform all of our white-box tests without changes to the design of our project. When using the Selenium test framework for blackbox tests, we encountered some problems that made tests fail as a result of slowly loading external resources. A number of measures have been put in place to mitigate the problems of loading external javascript files. First, the test will wait a longer period of time before clicking on items on the page. Second, if a test fails, the test will be restarted and tried again. Finally, all external resources have been made moved to the local project to speed up loading time. We also used the Bootstrap front-end framework to design our webpages. This framework was not adopted until Iteration 4. There were some minimal changes to the the pages found in the /public directory in order to accommodate for this framework. For example, each of the pages in the /public directory needed to be organized into rows and columns in order to gain the benefits of Bootstrap's responsive design. Other than this, there were significant design changes when using the Bootstrap framework.

Future Plans

If we had enough time to fully develop the project, we would have liked to communicate with actual daycare professionals and day care companies. We would have liked to collect actual information about how they would use *CheckinChildren* modify our project accordingly. We would also like to collect information about how our system would be better than the existing system that the daycare currently uses.

We would also like to implement a full Android/iOS app to accompany the web service. This would include push notifications for parents and more streamlined use by employees.

Personal Reflections

Harsh I thought the process we implemented went smoothly. We did not have any major conflicts that arose during the project. Whenever a small issue came up, the person with the problem would send out a message over Slack and the group would have the problem resolved in 15 minutes. I learned a lot about how to keep a team organized and updated. Without the ability to quickly communicate with the team, I believe we would have been much more disorganized and less capable.

Alex This project and process turned out a lot better than I expected! From throwing around ideas, to getting a team together, to actually seeing progress has been fantastic. When compared to being handed a project, I found creating a project from scratch to be a lot more rewarding. The planning, teamwork, and execution were a lot more involved and fun. There was a lot more discussion regarding how and what features to implement versus last semester and that kept our group meetings interesting. My group members were also great to work with, as we were usually on the same page and everyone had something to contribute. Overall, it was a great experience working on a project that started out as a simple idea of “child safety” and watching it grow.

Nick I thought this project went great. While I was not very invested in the idea, the fact that I got to contribute to what our project did certainly helped it stay interesting compared to being handed a project like last year. On top of that I think the team worked well together. Everyone was almost always on time to meetings and presentation. Additionally, everyone communicated well with each other when they had problems and were willing to help each other out, and our group always had a healthy mix of joking around and getting things done. Overall, I think this project was a great experience and am proud of how much we did this semester and how functional the idea became.

Olzhas My teammates made this project go smoothly. They are all really responsible students and always willing to help each other. For the first time I took part in project that was built from scratch. Great experience.

Elzabad The project went extremely well and I had an enjoyable time working on it. Everyone in the group were focused on the goal that we planned and gave it a hundred percent. The process that we decide on work really well for our team. Everyone was able to fully embrace it. It was a great experience and I feel better prepared for working in software as a result.

Matt This semester's project was a good experience for me working with a group to build a full web service from scratch. It presented many challenges and tasks to accomplish, which always kept us interested and engaged. The team worked well together, and our Agile/XP-like process was beneficial in keeping things organized and tracking our goals. In the end, I am satisfied with our final product. Our process was smoother than last semester and led to a better overall experience. We were better at communicating, documentation, and testing throughout. This helped alleviate the stresses of branch merges, conflicts, and other issues that inevitably arise in any group programming effort.