Stimulus Package

Version 2

Dillon Gee | Jazmynn Daos | Kyla Lamontagne | Vamsi Koduru | Wilbur Chen

Table of Contents

Revision Log

highlight = change from Version 1

Business Case:

Edited *What Are We Building?* section to add to features list

Requirements Document:

Added to and deleted from (because no longer implementing moderators; this is in exchange to customizable accounts) the functional requirements
Deleted from nonfunctional requirements (because no longer implementing moderators; this is in exchange to customizable accounts)

Design Plan:

No change

Project Plan:

Updated to include task lists that were used for sprints and execution tests

Test Plan:

No change

Keystroke Analysis:

No change (new document addition from Stimulus Package 1)

Communication Covenant:

No change

Combined Gantt Chart:

No change

Business Case

What Are We Building?

We are extending the Anki flashcard software by building a web application for sharing cards within the UCI School of Medicine. The Anki application currently is used to create, review, and share flashcards but the sharing feature is not entirely user friendly. Our web application will allow users to share their flashcard decks with one another more efficiently (horizontal integration). The users (students of UCI Medical School) will upload flashcards they have made themselves onto the central database into many shared deck based on subject. Users will then be able to view all of the cards within the shared decks. Users are able to edit cards, and then download the deck with those card edits. We will also implement some sort of customizable user account features that will make each login session per user more customized to their class year (MS1, 2, 3, or 4).

Why Are We Building It?

Building a website to compliment the Anki flashcard mobile application will create a more connected learning community within the UCI Medical School students and will improve the application's user experience.

The Anki desktop application allows users to create personal decks and share them with other Anki users via import and export. The deck-sharing feature requires users to manually import and export each flashcard deck from the AnkiWeb to use and edit other's decks, putting users off from using the sharing feature. Anki flashcard decks are mainly individual, and this causes current students to do a lot of redundant work. If they were all able to add and edit to a shared deck they could spend more time on studying the best definitions, and making cards better, rather than spending time to just make the cards.

UCI Medical School wants to create a community for the students and improving the sharing feature will allow this. By building out these sharing capabilities into another website, we are able to control who has access to the decks (only UCI Medical School Students) so as to keep UCI a competitive school while creating a community learning experience.

Requirements Document

Functional Requirements

- Users can register for an account, login, and logout.
- Users can upload files to the system.
- Users can delete files from the system.
- Users can download files from the system.
- The upload/download page automatically refreshes with an updated download list once a user uploads a file.
- Decks will be searchable for filter fields (Title, Professor, Topic, etc.).
- Users can view the contents of a deck (overview of all the individual cards).
- Users can view an individual card of a deck.
- Users can edit a card's content (front, back) and the edit will be updated into the database so that users can then download the deck with the added edits.
- Main database will automatically update with any changes (add approved new cards for edited cards, add new cards that were uploaded).

Non-Functional Requirements

- Website access is restricted to registered users only.
- System notifies user about any actions (i.e. "You are now logged in as...").
- File uploads are restricted to apkg files only.
- Available download files appear as a list of links of uploaded files.
- Website will be user-friendly.
- Website design and aesthetics will be simple and clean.
- System will display errors to the user when appropriate (i.e. invalid user, wrong password, upload error, etc.)

Design Document

Programming Languages
- Ruby

Frameworks
- Ruby on Rails

Other Libraries and Dependencies
- Gems, applicable to our project (Devise, Filterrific)

Overall Architecture
- MVC

Data To Be Stored
- User account information: usernames, email, passwords, upload history
- Flashcard decks in .apkg format
- Flashcard edits
- Deck class/category name organization structure
- User activity log: username, timestamps, action

Database
- SQLite

Software Components
- Model:
    - Decks
        - Methods: Download, Upload, Change Privacy Settings, Sync with central database
        - Attributes: Number of cards, tags, sub decks, master decks
- Accounts
    - Privileges
- Flash Card
    - Attributes: Edit, Remove, Deck Assignments
- Controllers:
    - View Cards:
        - Methods: View All, Filter view, Search
    - Accounts:
        - Methods: Log-in, Log-out, Sync, Share, Download, Edit
- View:
    - Web browser interface

Project Plan
Fall 2014: Week 1 – Finals Week

Week 1-3 Task List (from Sprint 1)

- Switch database from MySQL to SQLite
- Start implementing deck categories/filtering
- Research .apkg and .file files

Week 4 Task List (from Sprint 2)

- Continue implementing deck categories/filtering
- Research .apkg and .file files

Week 6

- Finish UI mockups
- Continue implementing reading .apkg and .file code on developmental build; back-end code
- Start gathering methods for user testing plan

Week 5/6 Task List (from Sprint 3)

- Fix search algorithm for filter
- Start with unzipping of anki files
- Start implementing new sort interface

Week 7

- Continue implementing reading .apkg and .file code on developmental build; back-end code
- Start implementing UI design (from mockups) on developmental build
- Refine methods for user testing plan

Week 8

- Continue implementing reading .apkg and .file code on developmental build; back-end code
- Continue implementing UI design (from mockups) on developmental build
- Finalize methods for user testing

Week 7/8 Task List (from Sprint 4)

- Create user communication system
- Follow-up with Kevin on user sign-up email
- Check/figure out Google analytics,
- Add error message for invalid login
- Clean up file management
- Add logo onto the filter/home page
- Clean UI design, phase 1 (consistency)

- Create view for individual cards
- Drop down for classes in filter page

Week 9

- Continue implementing reading .apkg and .file code on developmental build; should have some type of “viewable” deliverables by now
- Continue implementing UI design (from mockups) on developmental build
- Finalize methods for user testing
- Start finding potential users for user testing

Week 9 Task List (from Sprint 5)

- Finalize user testing incentives with Kevin
- Create final user testing plan docs
- Coordinate with coders to update community page
- Fix homepage UI design
- Continue working on view card (image problem)
- Change file management
- Implementation: Repackaging/rezipping
- Research: Repackaging/rezipping

Week 10

- Finish implementing reading .apkg and .file code on developmental build; should have some type of “viewable” deliverables by now
- Finish implementing UI design (from mockups) on developmental build
- Continue finding potential users for user testing

Week 10 Task List (from Sprint 5)

- Conduct user testing sessions with medical students
- Compile user testing results
- Analyze user testing results
- Create new tasks from user testing results
- Continue changing file management system
- Continue working on repacking/rezipping decks
- Continue fixing image problem on view card
- Continue working on view cards of certain decks (general)
- Continue implementing design UI fix-ups
- Create mock-ups for new UI design ideas
- Create final demo presentation

Finals Week

- Final demo and presentation for sponsors, professor, and TA

Test Plan

Software Testing Plan

Revision History
None

Introduction

Testing for AnkiShare is to be done with students who are currently enrolled in the UC Irvine School of Medicine. Several testing methods will be employed, namely heuristic evaluations, think-aloud evaluations, and cognitive walkthroughs. The goal of testing is to receive feedback from the testers that can be used to improve functionality and user experience. Constraints regarding user testing include a slight lack of technological background in medical students, as well as availability for testing sessions. It is assumed that students will be willing to participate in testing, and that they will be aiming to improve the software with honest feedback. Testing is dependent on students being willing to participate, as well as availability.

Test Item Pass/Fail Criteria
Use-Cases that will be tested include:

- Uploading a deck
- Downloading a deck
- Filtering based on Professor, Topic, Class
- Contacting the AnkiShare Team
- Making an Account

Uploading a Deck

- Success: If users can successfully upload a deck starting from the landing page without intervention.
- Fail: If users require assistance to upload a deck starting from the landing page, or if they spend more than 30 seconds.

Downloading a Deck

- Success: If users can successfully upload a deck starting from the landing page without intervention.
- Fail: If users require assistance to upload a deck starting from the landing page, or if they spend more than 30 seconds.

Filtering based on Professor, Topic, Class

- Success: If the user uses the filter function and the filter is applied with the appropriate criteria.
- Fail: If users require assistance to apply the filter or spend more than 15 seconds doing so.

Contacting the AnkiShare Team

- Success: If users can successfully write a message to the AnkiShare team without intervention.
- Fail: If users require assistance to contact the AnkiShare team, or if they spend more than 30 seconds doing so.

Making an Account

- Success: If users are able to successfully create an account and log in to AnkiShare, starting from the landing page.
- Fail: If users require assistance in creating an account and logging in to AnkiShare, or if they spend more than 30 seconds doing so.

Test Deliverables

Test Plan

(This document.)

Test Cases

None on file yet.

Test Scripts

None on file yet.

Defect/Enhancement Logs

None on file yet.

Test Reports

None of file yet.

User Testing Plan

Heuristics Evaluation

Our group will use the most up-to-date user interface mockups and will evaluate the mockups based on the ten basic heuristic principles. After completing the heuristics evaluation, the team will incorporate our findings (both changes and keeps) into our interface design implementation during development. We will update our mockups accordingly, and then continually re-perform the heuristics evaluation until the interface deems fit and meets all the heuristics principles.

Actual User Testing

Our testing users will be voluntary first-year UCI medical students. The methods used will be: cognitive walkthroughs, think aloud exercises, online surveys, and Google Analytics. For cognitive walkthroughs and think aloud exercises, one team member per user will be present during each session to observe users. For all methods, team members will take notes. After each session per method, all team member notes will be compiled into a major list to finalize points to implement in development. User testing will be conducted during both soft and hard releases of the website.

Keystroke Analysis

Introduction

User testing was done on location at the UC Irvine Medical School with four medical students. Participants were tested one by one, with one team member walking them through the test, and the other making observations and taking notes.

The testing method used was think-aloud testing, in efforts to receive feedback on AnkiShare and the user experience. All users were asked the same questions and followed the same task-list.

1. Tester A (Jessica V.)
    - Experienced AnkiShare user
    - Has already been uploading decks
2. Tester B (Taylaur)
    - Has used AnkiShare once
3. Tester C (Brian)
    - Experienced AnkiShare user
    - Has uploaded decks and reviewed the website in detail before
4. Tester D (Jessica G.)
    - Has never used AnkiShare

Tasks

1. Apply filters
    - Apply “Sattar” to Professor field and “Topic (a-z) to Sort-By field
    - Time limit: 30 seconds
2. Download Deck
    - Click “Download Deck” button for any given deck
    - Time limit: 10 seconds
3. Upload Deck
    - Navigate to upload page, fill in all the form fields, and submit the form
    - Time limit: 60 seconds

Results

Apply Filters

- Tester B selects “Professor a-z instead of “Topic a-z”
- Tester D puts “Sattar” under “Class Name”
- Both testers needed to be reminded of their errors
- Because the filter is applied automatically, they waited for the results, but they never came
- Testers B and D were both confused about where the “filter” was

Download Deck

- All testers were able to download a deck under the time limit

Upload Deck

- Tester D was unable to find the "Upload Deck" button, and it had to be pointed out
- Tester C was confused about what "Author" meant; whether it was himself or the author of the content of the cards (i.e. the author of the textbook that covers the material)
- Tester B successfully completed the tasks, but did so in a different order (not top-to-bottom)

Discussion

Apply Filters

Multiple testers were unable to find the filter, despite it being a large widget in the center of the page. One of them thought the fields were for the "Upload" function. While they liked the filter fields, several testers suggested adding a title "Filter" on top of the widget. Another suggestion that was received was for the "Sort-By" field to be moved from the widget to the actual list. This allows for better association with the list (users didn't know the "Sort-By" function was referring to the list).

Download Deck

All users were able to quickly locate the "Download Deck" button and download the deck. The color of the button helps in making it stand out among the rest of the deck information.

Upload Deck

One tester was unable to locate the button, but this could be because their window was scrolled down (the upload button was not on the screen). The button is relatively small, so they managed to miss it when they scrolled back up, so they scrolled down again. Perhaps this button could be enlarged. Once on the upload page, two users questioned the "Author" field, asking whether or not the "Author" was themself, or the author of the material that the cards covered. This could be clarified by changing the wording to "Author (Your Name)", or "Uploader". One tester commented that it was defined in the "Uploading Guidelines," but also noted that not many people would actually read it.

Additional Feedback

- Add the thin grey line under the logo on the Downloads page to the upload page as well
- Clarify the parameters for "Student Year."
    - Tester didn't know if he should put "2" or "MS2"
- Auto-filter is confusing, not sure if filter has been applied or not
- A preview deck function would be useful

- A tagging system would be useful
    - Some decks are tagged to several professors, topics, and textbooks
- Some sort of icons to indicate what “kind” of deck it is
- When switching pages, it shouldn’t bring you back to the top of the page
- Add more space for notes; testers wanted to type lengthy notes, but there isn’t space for it.
- Some sort of username/email/initials associated with uploaders
- Perhaps implement a year-setting, so the user would only be exposed to material from that year (MS1, MS2, etc.)
- Buttons for classes/topics instead of the entire list.
- Rating system for decks
- Switch “Class Name” to “Course Name”
- Actual button for Sign Out/Sign Up
- Show deck size
- Allow users to delete their own decks

Communication Covenant

Team Contact Information

Dillon Gee:
dgee@uci.edu
650-200-8842

Jazmynn Daos:
jdaos@uci.edu
949-422-8036

Kyla Lamontagne:
k.lamontagne93@gmail.com
408-430-6393

Vamsi Koduru:
vkoduru23@gmail.com
408-624-6598

Wilbur Chen:
wilburc@uci.edu
408-624-6598

Agreement Terms

Meetings

- The team will meet with each other in person for at least 2 hours each at least once a week outside of class. Each week will depend on the schedule and availability of each team member.
- The team will meet with sponsors at least once every two weeks (bi-weekly) and communicate with them via email every week.
- All team members will be expected to work individually on their tasks and responsibilities during non co-working hours.
- All team members are expected to be present at team and sponsor meetings, unless given advance notice.
- If a team member is late or missing from a meeting without notice, he/she will be expected to catch up with the team's progress. If it becomes consistent, the team will bring up their concerns to the professor and TA.

Communication

- All team members will use GroupMe, email, Facebook, and phone (call//text) to communicate with each other.

- All team members will check their GroupMe messages and emails daily.
- All team members will be expected to respond to all emails, GroupMe messages, texts, or phone calls, when applicable to them, as soon as possible.
- All team members will contribute equally to the project, and will deliver their responsibilities on time and professionally.
- All team members will regularly seek help from each other when needed.

File Management

- The team will use Google Drive/Docs to work on documents collaboratively, and will post all final versions on GitHub for the professor and TA. Project development (coding) will take place on GitHub.

Spring to Fall

Last Spring it was very easy for our team to meet with each other at least twice or three times a week because all of our schedules were flexible and/or free, so we were able to set meeting times in stone. This quarter, it's going to be difficult for our group to meet as often since our class and work schedules all conflict with each one another. Because of this, it will be important for our team to make sure that as individual team members we execute our responsibilities on our own so that our project as a whole stays on track. Communication and motivation to do work alone are going to be key for us this quarter since we won't be able to co-work with each other as often. Overall, our team worked well together last Spring and we have been continuing to do so this quarter as well.

Combined Gantt Chart

Gantt Chart

(See chart attached at the end of the document.)

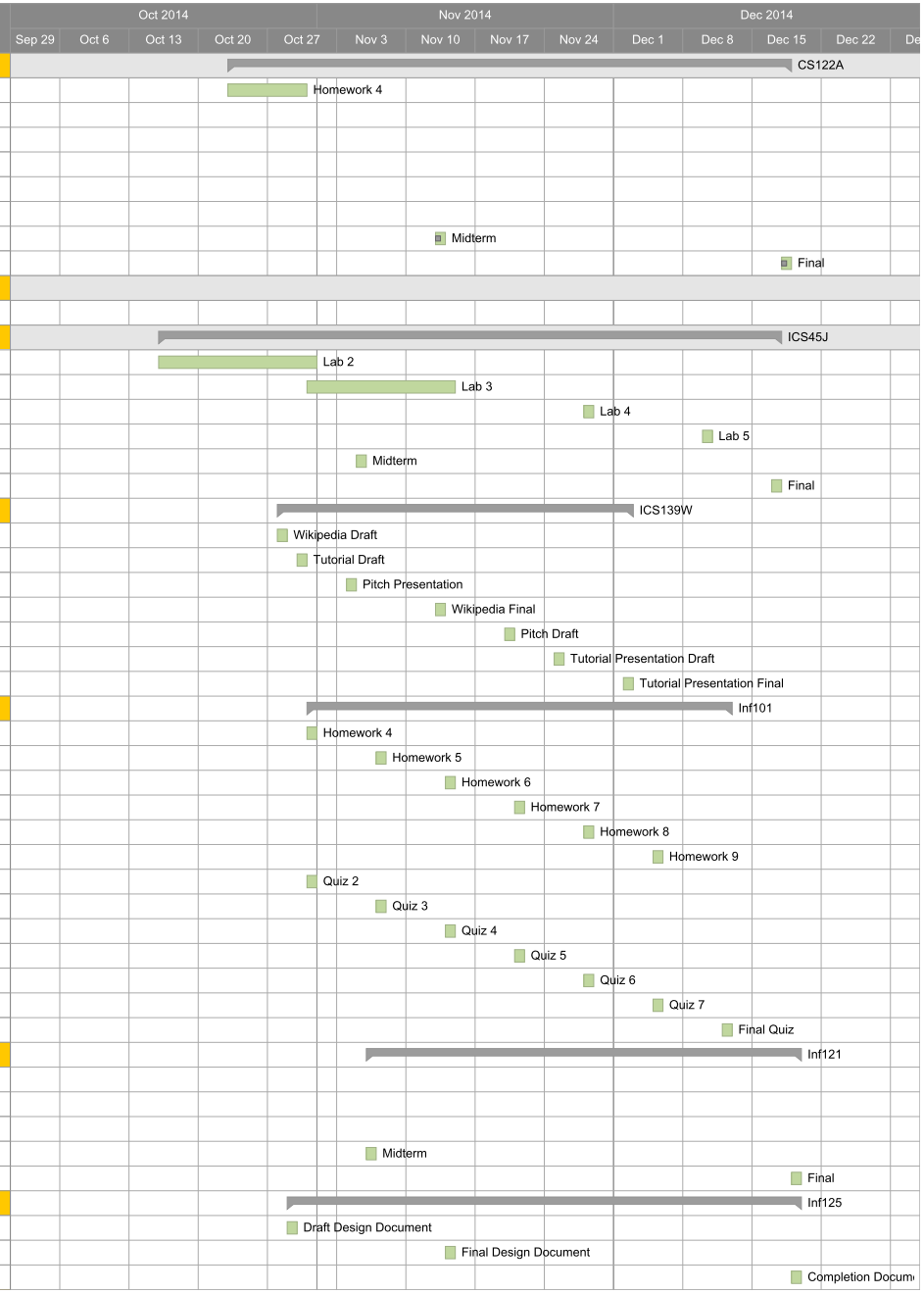
Project Plan Implications

Each member of our team is taking at least 12 units (and up to 20 units) this quarter, and all of us have at least one other project course. Based on our Gantt chart, each team member will have a lot of other work on their hands (both class and work wise), and possibly even more than expected since not all of our classes have up-to-date definite schedules. For 191B this quarter, much of the work will have to be done individually (in that we can't all be present together at the same time) and each individual will be responsible for completing his or her deliverables on time (be it hard assignment deadlines or coding development on the website). Our team will still make it a point to meet with each other, but won't be able to meet as an entire group with everyone present as often as we were compared to last quarter.

Inf191 Combined Gantt Chart

AnkiShare: Dillon Gee, Jazmynn Daos, Kyla Lamontagne, Vamsi Koduru, Wilbur Chen

D	J	K	V	W	Task Name	Start	End	Duration	Notes
D	J				CS122A	10/23/14	12/18/14	41	
					Homework 4	10/23/14	10/30/14	6	
					Homework 5				Not specified
					Homework 6				Not specified
					Homework 7				Not specified
					Homework 8				Not specified
					Mini Project				Not specified
					Midterm	11/13/14	11/13/14	1	
					Final	12/18/14	12/18/14	1	
D					CS171				
					Insert here				
	J				ICS45J	10/16/14	12/17/14	45	
					Lab 2	10/16/14	10/31/14	12	
					Lab 3	10/31/14	11/14/14	11	
					Lab 4		11/28/14		
					Lab 5		12/10/14		
					Midterm	11/05/14	11/05/14	1	
					Final	12/17/14	12/17/14	1	
				W	ICS139W	10/28/14	12/02/14	26	
					Wikipedia Draft	10/28/14	10/28/14	1	
					Tutorial Draft	10/30/14	10/30/14	1	
					Pitch Presentation	11/04/14	11/04/14	1	
					Wikipedia Final	11/13/14	11/13/14	1	
					Pitch Draft	11/20/14	11/20/14	1	
					Tutorial Presentation Draft	11/25/14	11/25/14	1	
					Tutorial Presentation Final	12/02/14	12/02/14	1	
			V		Inf101	10/31/14	12/12/14	31	
					Homework 4	10/31/14	10/31/14	1	
					Homework 5	11/07/14	11/07/14	1	
					Homework 6	11/14/14	11/14/14	1	
					Homework 7	11/21/14	11/21/14	1	
					Homework 8	11/28/14	11/28/14	1	
					Homework 9	12/05/14	12/05/214	1	
					Quiz 2	10/31/14	10/31/14	1	
					Quiz 3	11/07/14	11/07/14	1	
					Quiz 4	11/14/14	11/14/14	1	
					Quiz 5	11/21/14	11/21/14	1	
					Quiz 6	11/28/14	11/28/14	1	
					Quiz 7	12/05/14	12/05/14	1	
					Final Quiz	12/12/14	12/12/14	1	
	J				Inf121	11/06/14	12/19/14	32	
					Design Studio 2				Not specified
					Design Studio 3				Not specified
					Design Studio 4				Not specified
					Midterm	11/06/14	11/06/14	1	
					Final	12/19/14	12/19/14	1	
D				W	Inf125	10/29/14	12/19/14	38	
					Draft Design Document	10/29/14	10/29/14	1	
					Final Design Document	11/14/14	11/14/14	1	
					Completion Document	12/19/14	12/19/14	1	



D	J	K	V	W	Task Name	Start	End	Duration	Notes
		K	V	W	Inf133	10/29/14	12/19/14	38	
					Codecademy jQuery	10/29/14	10/29/14	1	
					AJAX Quiz	11/07/14	11/07/14	1	
					Location Quiz	11/07/14	11/07/14	1	
					Mid-term Evaluation	11/07/14	11/07/14	1	
					Real-time Web	11/14/14	11/14/14	1	
					Real-time Map	11/21/14	11/21/14	1	
					Multi-touch Assignment	12/01/14	12/01/14	1	
					Quiz on HCI Ubicomp	12/05/14	12/05/14	1	
					Android Programming Assignment v1	12/12/14	12/12/14	1	
					Final Class Evaluation	12/19/14	12/19/14	1	
					Shaping Things Quiz	12/19/14	12/19/14	1	
					Android Programming Assignment v2	12/19/14	12/19/14	1	
D	J	K	V	W	Inf191B	10/29/14	12/19/14	38	
					Final Novemberfest Poster	10/29/14	10/29/14	1	
					Second Draft of Test Plan	10/30/14	10/30/14	1	
					Sprint 2 Report	10/31/14	10/31/14	1	
					Sprint 3 Report	11/14/14	11/14/14	1	
					Sprint 4 Reports	11/28/14	11/28/14	1	
					Execution Tests 4	12/02/14	12/02/14	1	
					Progress Report 4	12/03/14	12/03/14	1	
					Peer Review 3	12/04/14	12/04/14	1	
					Stimulus Package 2	12/12/14	12/12/14	1	
					Sprint 5	12/12/14	12/12/14	1	
					Peer Review 4	12/19/14	12/19/14	1	
		K			PH166	10/29/14	11/13/04	23492	
					Homework 4.1	10/29/14	10/30/14	2	
					Homework 4.2	10/29/14	10/30/14	2	
					Homework 5.1	11/05/14	11/12/14	6	
					Homework 6.1	11/12/14	11/13/14	2	
					Homework 6.2	11/12/14	11/13/14	2	
					Homework 7.1	11/12/14	11/13/04	23482	
					Homework 8.1	11/19/14	11/27/14	7	
					Homework 8.2	11/19/14	11/27/14	7	
					Homework 9.1	11/26/14	12/03/14	6	
					Homework 9.2	11/26/14	12/03/14	6	
					Homework TBD	12/03/14	12/04/14	2	
					Case #1	10/29/14	11/10/14	9	
					Case #2	11/12/14	11/24/14	9	
					Final Exam	12/17/14	12/17/14	1	
				W	PS21A	10/29/14	12/25/14	42	
					Presidential Honeymoons	10/29/14	11/04/14	5	
					The Pollster's Puzzle	11/07/14	11/13/14	5	
					Presidential Elections	11/19/14	12/25/14	27	
					Response Paper	11/25/14	12/11/14	13	
	J				S135	10/30/14	12/18/14	36	
					Midterm Exam	10/30/14	10/30/14	1	
					Critical Paper	12/04/14	12/04/14	1	
					Final Exam	12/18/14	12/18/14	1	
		K			Work	10/26/14	12/19/14	41