Software Lab EE461L

Team A7: Board Game Database

TEAM A7

10.09.2020

Cedrik Ho, Allegra Thomas, Grant Ross, Sanne Bloemsma

INFORMATION

GCP Deployed App: https://fall-2020-ee461l-teama7.uc.r.appspot.com/
Github repo link: https://github.com/UT-SWLab/TeamA7
Team Canvas group ID: A7

Team Members:

Allegra Thomas (at35737)
Email: allegrathomas@utexas.edu
GitHub: AllegraThomas

Cedrik Ho (ch45935)
Email: cedrikho@utexas.edu
Github: CedrikHo

Sanne Bloemsma (scb2936)
Email: sbloemsma@utexas.edu
Github: sbloemsma

Grant Ross (ghr344)
Email: ghross@utexas.edu
Github: Grant-Ross

MOTIVATION AND USERS

The motivation of this project was to create a cohesive web application for users to browse board games and be able to find out more about board games, their publishers, and various board game genres. It was to some degree inspired by various board game hobby pages. However, it is our hope that this web application will be more intuitive to use and be less fragmented in terms of genres and brands of board games hosted.

We believe that this web application will be useful to a wide range of demographics and be particularly useful to individuals who regularly play board games with others. We anticipate that access to this website will also vary from mobile phones, tablets, personal computers, and other various devices with internet connectivity.
Thus ease of use and flow of information is highly important to the team.

USER STORIES

Our user stories were all hosted on Github. The following link will show you each of the user stories created and the state it is in.
https://github.com/UT-SWLab/TeamA7/projects/1?fullscreen=true

USE CASE DIAGRAM

DESIGN

The design of this website includes a home page with general information about entries in our API, an about page with information about the website itself, and 3 list pages for each of the 3 models in our database. There is also a header on each page that the user can use to get to any of these pages so that every other page is readily available.

- UML

MODELS

Board Games: This model contains names, general information, and gameplay descriptions of various board games. Board games are published by publishers, and can be categorized into one or more distinct genres.

Publishers: This model contains the names, general information, and history of various board game publishers. These publishers produce and sometimes design different board games, and publishers often specialize in publishing games under one or more genres.

Genres: This model contains names and descriptions of various established genres of board games.

TOOLS AND FRAMEWORKS

The main frameworks for this project are as follows:

- BootStrap4 - used in **Phase I**
    - Used to to create basic CSS structure and mobile development.
- GCP (Google Cloud Platform) - used in **Phase I**
    - Used to host the deployed site and also a third party cloud storage solution for code and future data
- GITHUB - used in **Phase I**
    - Used for version control and team management. Used as a host for issue tracking and User stories.

PHASE I REFLECTION

This initial phase was a large first step in developing a cohesive and modular web application. The team as a whole was firm in making decisions and few conflicts arose.

As a team we knew coming in that none of us had much experience with a larger web application. Therefore early on it was not too surprising that we had trouble with creating a solid basic framework with correct inheritance. Specifically we had issues with extending templates without overriding existing content and achieving consistency of grids and CSS elements across model pages. To solve these issues, we re-worked the Flask framework from the Bootstrap4 tutorials and created a default CSS sheet.

We struggled early on with creating a basic file that could easily be given to each of our models. This lack of planning caused some blockers for other team members early on. Once we adopted a more religious use of Githubs user stories and a true Scrum asking the three key questions, these issues were immediately solved. There were also some unexpected issues in terms of elements behaviours and resizing when the page was not at its default width. We created a larger template and standardized cards width and height to create the proper ratios and re grouped data so that growth of the page was as expected.

When deploying the app there was some confusion amongst naming of the python file, so after weeks of using “app.py” as the name of our file in order for it to properly deploy on GCP and be compatible with the “.yaml” file it had to be renamed to “main.py.” Although it caused some confusion initially it was easily resolved after reading some documentation for GCP.

Overall, I believe that due to our lack of experience in web development, we struggled to predict certain issues ahead of time.The rather quick and responsive nature of web applications is a new area of software development that the team as a whole has had to get adjusted to. We feel that our use of github issue tracking and clear user stories were very beneficial in helping create a quick and straightforward process for development goals. We are excited and believe that Phase II will be even expansive and rich in information and features.

Despite the challenges that arose, we were still satisfied with some aspects of our implementation and workflow. Our weekly stand ups helped us track progress and stay accountable, and our fair breakdown of work based on the models chosen helped us accomplish our goals in parallel. We believe that each individual will be considered the “manager” of the model they worked on in Phase I, since we found a clear assignment of responsibility for a specific model greatly improves communication and productivity. Also, interactions with API’s went well, as we have registered our company with the API providers and access to the API’s seem to work fine. While we did hard code information, we see a clear path to filling out all three models using our APIs.

As stated earlier our team work structure and communication has been a source of pride and success for the team as a whole. We believe that our team contract has been enforced and integral to our quick resolutions and continued respect for each other's ideas and work. Overall, this reflection has been greatly beneficial in helping the team analyze the issues we have overcome and the weaknesses we still hold going forward into phase II.

Five Things We Struggled With:

1. Achieving consistency across model pages
2. Extending templates without overriding existing content
3. Adjusting page content with page resizing
4. Using one virtual environment between team members
5. Predicting issues ahead of time

Five Things That Worked Well:

1. Dividing work between models
2. Getting information from sources with APIs
3. Setting up the website using the Bootstrap tutorial from class
4. Syncing code with GitHub
5. Helping each other with code issues in meetings

PHASE II REFLECTION

This phase of the project was much more difficult for us. While the requirements had risen in complexity from the first phase, we also struggled due to a combination of poor planning, a lack of communication, and procrastination.

After our success in Phase 1, our team was ready and eager to begin planning for Phase 2. This initial planning was brief, however, and our proposed division of the work was not fleshed out enough, as the design of our project going into Phase 2 was not fully understood or unanimously agreed upon by our team. This meant that while we had a general idea of who would be working where, the specific files each member would focus on was unclear. Another thing we did not plan well was the implementation of the database. We implemented one of our models early on, but the other two were added way too late. We failed to notice that one of the APIs we had intended to use for one of our models limited usage from unverified organizations, meaning it was no longer a feasible option for our project. We quickly found another API that suited our needs and in fact added a more pleasing visual aspect to the site. Though as a team we agree this was unnecessary last minute stress, and a more rigorous research plan and earlier implementat would have been less stressful for all parties.

We believe that we should have focused more on creating the databases early on so that they could be seamlessly implemented as we built more pages of the site. What we ended up doing instead was build pages to get their information without their specific collection, then replaced that method with calls to the individual collections as they were being made. In the end, our project required re-design of database access and at times certain team members were blocked by the lack of fully implemented Database collections.

Our initial Database collection and REST-API usage for a single page worked well, but it failed to scale horizontally. As we added more information, broken links occurred as did confusion on what sub fields from Database objects should we be accessing. This issue was solved by segregation collections and rerouting REST_API calls for all 3 modules separately. While this is a clean and easy solution, it did create more methods and possible app routes to test.

We believe that more frequent checkpoints with smaller goals will help us in Phase 3.

Monitoring user stories more frequently will also help. We believe that our discussion on the technical aspect of implementing the user stories needs to be cohesive and in depth.

Despite the challenges that arose, we were still satisfied with some aspects of our implementation and workflow. We were able to satisfy the project requirements and communicated well near the end. The information and code we wrote this time around was much more standardized and we believe it will help us a lot in phase 3. Our Testing is simple but we believe it is effective and covers the necessary requirements for the stage of development we are in. We will be much more diligent and communicate more often for phase 3. We look forward to continuing the maintenance and build of this project.

Five Things We Struggled With:
Database management
Scaling implementation
REST_API call structure
Appropriate API calls
Communication of division of work

Five Things That Worked Well:
1. Documentation of Issues on Github
2. Quick transition to find new API's
3. Syncing code with GitHub
4. Testing framework
5. Teamwork and communication in the later part

CITATIONS

What sources did you use? Provide links to tutorials, books, Stack Overflow pages that you used and indicate how each source was used.

"BootStrap Tutorial." Bootstrap 4 Tutorial, www.w3schools.com/bootstrap4/default.asp.
"Web Development 101." The Odin Project
www.theodinproject.com/courses/web-development-101/
"Google Cloud Platform Tutorial" GCP
https://console.cloud.google.com/getting-started?tutorial=toc

Testing Overview and Goal

This section outlines the key areas of testing for each phase of TeamA7's EE 461L Project. For each Phase of the project the team will outline what user story criteria we would like to test and how we implemented that testing. Testing for our project is cumulative, and this section currently holds testing for Phase II. It is our hope that this documentation helps the group identify areas that need to be tested for each phase as well as communicate our ideas on testing to not only the group, but other interested parties.

Scope

The scope for testing in Phase 2 is to test the fundamental transition of pages and elements on our webpage. We believe that this is the most important part to test as the main purpose of the site is for it to be a functional resource for users. The rest API calls for the most part were debugged when we populated the database, and based on the current information we are using from the APIs we don't believe they should be the focal point of our testing for this Phase. In later Phases we will implement some standard patterns for checking information from our API's into our DB to reduce duplication of elements or unwarranted data dumps.

Objectives

For testing our site, we want to ensure we maintain an aesthetically acceptable and functional site that allows users to navigate between our three models and instances of those models without issues. For testing in Phase II, our focus is predominantly front end and focused on the user experience of the site. Our MongoDB database and the calls we made to our APIs have repeatedly been tested in early stages manually (printing to the console to compare) as well as automated (checking the information we added to the database was also in the API we pulled from).

Scalability

As the project grows and we contribute more information to each of our models, we will have to create more tests. The goal is for our tests to be cumulative, for each Phase we will not have to eliminate old tests and can just build on what we have already verified.
Code Coverage: For code coverage, we currently have methods in main that are not called in the current stage, but in further additions to our models will be used. At the moment testing does not cover those methods.

Phase II Testing Breakdown

Front End Testing Goals (Based on User Stories)

- ❏ All main navigation bar pages are accessible from anywhere on the site
- ❏ All list pages load at a reasonable speed and accurately
- ❏ Pagination functions as expected
- ❏ All instances of a model shown in list pages link to an instance page
- ❏ Instance page links link to related links of each model
- ❏ Image and window resizing doesn't compromise the readability of the site
- ❏ Frequently clicked elements of page(Carousel, Nav Bar) do not lose functionality with repeated usage
- ❏ [Phase II] no external sites aside from about page links to sources

Testing Implementation

Front end testing can be rather difficult to automate fully, particularly when it comes to visuals and aesthetics. The team as a whole spent hours going through links, pages, and buttons in an attempt to find flaws or odd issues with the site. The issues if not instant fixes were logged into our Github issues tracker.

Below are a few issues we found during testing:
Automated Testing (Selenium, Flask library)

- The genres list and publishers list pages were initially modeled off of the games list page. The title name of the genres list page was the same as the Board Games List page

Manually

- Content was not properly generating on instance pages for publishers
- Links to instance pages from the genre list page were broken
- Typos regarding singular and plural of model names (genres vs genre, games vs, games, publishers vs. publisher) which yielded jinja errors
- Null publishers in the board games collection (API to Mongodb storage issue)
- Refreshing an instance page of a genre or publisher, the relevant games links disappeared
- Generally visual issues with image and text resizing which were more difficult for out automated testing to detect

Testing Frameworks

- Selenium
- Flask Test Library

Homework 3: Team 7 Contract

Team Members: Cedrik Ho, Allegra Thomas, Grant Ross [Phase I Lead], Sanne Bloemsma

In this document, we will establish our expectations for team communication, conflict resolution, and task allocation, as well as detail the strengths and working styles of each team member.

Team Communication

Our team's primary forms of communication will be Slack and Zoom meetings. We will use GitHub, integrated with our Slack channel, for code management. Our team will meet through Zoom twice a week, from 3pm-5pm on Wednesdays and 10am-12pm on Fridays. During meetings the team will define clear goals for each team member to complete before the next meeting. We will also hold daily standup meetings via Slack, in which each team member briefly details what they have completed since the last meeting, what they will complete before the next meeting, and any problems they have with their current tasks.

Team Strengths and Working Styles

For the team's strengths and working styles, we have only met once so far and therefore are not familiar with each other's abilities yet. We have identified our own strengths and working styles as well as where we may work really well together or may have points of conflict.

Strengths

Cedrik - Python back end testing and frameworks. Open to any task.

Allegra - Document editing, self-guided learning and research. Familiar with Python and Java.

Grant - Giving constructive feedback. Familiar with Java and Python. Very familiar with Github.

Sanne - Self-motivation, self-guided learning, and following instructions. Familiar with Java.

Working Styles

Cedrik - Clear deadlines and expectations. Individual technical work but frequent team meetings to go over all aspects of the project. Favors a clear hierarchy of task importance and order of completion within personal tasks as well as team tasks.

Allegra - Break tasks up into smaller parts and set a schedule for task completion. Complete individual work in chunks with clear goals and deadlines and get feedback from the team before moving on to the next chunk.

Grant - Starting a project early, working on it in small bursts leading up to the deadline. Make sure each member has a clear idea of what they are working on in order to avoid issues caused by lack of communication. Enjoys paired-programming occasionally, but only for larger, more complex tasks.

Sanne - Working on tasks for one to two hours at a time and compiling a project over a set period of time. Prefers to by the person typing while the other person observes and gives feedback or suggestions.

Conflict Resolution

Since conflict usually stems from a disagreement on how something is going to be done or was supposed to be done, we will be explicit on what tasks we are working on each week, who is responsible for that task, and what all team members expect from that completed task.

If a team member disagrees with actions taken or not taken by another team member, the former will initiate a private conversation with the latter to discuss a compromise. If this meeting doesn't resolve the situation, then the team member will discuss the issue with the rest of the group. If a team member disagrees with the direction, actions, or lack of action by the group, they will address their concerns at the beginning of the next meeting in a constructive manner.

For general decision-making, we will adhere to a strict all-or-none voting system and three-strike disciplinary system to ensure order. If a single team member does not agree with a decision, the proposal cannot pass. When this situation arises, everyone will collectively work together to find a compromise the team can progress with. Additionally, since there is no hierarchy within the organization, discipline for poor communication will be handled using a three-strike system. Actions that will receive a strike include failure to have work ready by a clear deadline and missing a meeting with no advance notice. On the third strike, we will communicate these issues with the professor or a TA to resolve the problem.

Task Allocation

Task allocation will change throughout the course of the project, but initially we have decided to break up the project and responsibility as follows:

-Documentation and Deliverables [All]

-Testing and Quality Assurance [Cedrik Ho]

-Front End UI UX [Grant Ross]

-Data Management, Manipulation and Metrics [Sanne Bloemsma]

-API Calls and DB Management [Allegra Thomas]

Expectations of Team Members

- Be responsible and familiar with their own workload and capabilities outside of this project and this class
- Complete tasks and portion of assignments as promised, both in respect to time and quality
- Be comfortable with voicing their concerns or suggestions for improvement when it comes to the project or pair programming
- Be comfortable taking criticism
- Be understanding of others with less experience who may not be as familiar with the information needed to complete portions of the project
- Be understanding that others' experience does not mean they are responsible for the majority of the project and they are responsible for making sure they do their fair share of work