Capstone Final Report: Agora

Kiara Barias
kiara.barias50@myhunter.cuny.edu
CUNY Hunter College

Chun Nam
chun.nam50@myhunter.cuny.edu
CUNY Hunter College

Gillian Pantaleon
gillian.pantaleon30@myhunter.cuny.edu
CUNY Hunter College

ABSTRACT

Agora is a web-based application with the purpose to serve as a resource for community building within commuter-based colleges, such as the CUNY schools. The idea came about by observing the lack of student life activity at Hunter College and the limited space on campus. The concept for our application is similar to the Events feature on Facebook. Agora enables users to organize and respond to gatherings in the real world. The application will let users schedule study sessions, find or provide tutoring, exchange textbooks, and/or create or join a meetup. Unlike existing solutions, Agora suggests convenient meeting places depending on the user's needs. Agora is intended for a democratic society where every member accessing the resource is expected to respect each other and police the use by an honor system.

ACM Reference Format:
Kiara Barias, Chun Nam, and Gillian Pantaleon. . Capstone Final Report: Agora. In . ACM, New York, NY, USA, 9 pages.

1 INTRODUCTION

The name of our project is *Agora*, which is pronounced 'ah-go-RAH'. The term refers to a public space used for assemblies and markets in ancient Greece; e.g. where the philosophers[11] met to have their discussions. The motivation behind Agora, comes from there being a limited number of resources for commuter-based colleges, these resources being outdated, and there being a diverse student body. This application aims to provide a holistic interface for students to find, suggest, and access resources at their fingertips.

(1) **Limited resources of commuter-based colleges**
In commuter based colleges such as the City University of New York (CUNY), there is very limited space for students on campus as well as outdated methods of communication. Space limitations restrict where students can study or hang out on campus. Additionally, ineffective communication channels such as flyers and mailing lists are used to make announcements. The current methods of communication make information difficult to find.

(2) **Diverse student body**
The student body of CUNY schools comprises of full-time, part-time, and non-traditional students. Thus, schedules vary significantly among students, which makes it challenging for them to find time to work on a group project, schedule a study group, have a club meeting, or to hang out. Failure to resolve this issue has resulted in a lack of community.

(3) **Outdated Resources**
School websites, such as Hunter College's Website [7], remain outdated. Although the college website is currently undergoing major improvements, the student life section contains outdated information as well as other sections. For example, the club listing contains organizations that are inactive or have dissolved. Also, student resources are placed in different areas. Each tutoring center has a website [4]. Due to resources being unorganized, the website is not user-friendly [3]
.

2 EXISTING SOLUTIONS

There exist some applications that offer solutions to the issues that motivated the application, Agora. However, these applications do not offer the complete solution that Agora strives to provide.

(1) **Student2Student[15]**
This application provides a platform for students to directly exchange textbooks with one another on campus. However, it does not suggest places for the transaction. Providing a location is unique to Agora.

(2) **Meetup[9]**
This application assists the user in creating and/or joining groups, but it does not assist in finding the most ideal place to meet based on past experiences, unlike Agora.

(3) **Facebook Events[5]**
This feature on Facebook makes the task of creating events easy. However, it does not provide helpful information on the location. For example, if student users are interested in creating a study group session, they have no way of knowing if the area is good for groups

, ,

or has free WiFi. Agora, on the other hand, provides such useful information and incorporates a rating system for each location.

(4) **Eventbrite[8]**
This application is a platform for large-scale event organization. It is catered towards businesses and organizations to assist in event planning. It is not appropriate for our target user, which is a college student. Agora meets the needs of college students by connecting them with fellow students and improving student life activity.

3 PROPOSED SOLUTION

We believe Agora has significance outside of the classroom. The application solves a real-world problem and may be a useful platform for the CUNY student body by providing the following features:

(1) **Textbook Exchange**
Students can buy, sell, donate, rent, or borrow textbooks from each other through direct communication. The application connects students and provides a convenient time and place for exchanges to occur. All transactions take place offline (i.e. outside of the app). This helps to replace services, such as Amazon Rentals, where students would have to place an order, wait for the package to be shipped, and then have the textbook potentially lost in delivery.

(2) **Tutoring**
Students can find tutors for specific courses and reserve sessions and/or provide tutoring services. This can help students find additional help for specific subjects at certain times that the university itself might not provide. Tutors might also be able to profit off of their tutoring sessions, thus potentially providing a source of income for students.

(3) **Study Group**
Students can create their own study groups with a designated time and place that works for them. They can also join their friends' group(s) or search for open study sessions and make new study buddies. This provides a sense of freedom for students to organize sessions based on their schedules that would be difficult not given this platform. This can easily be expanded to including clubs and special interest groups as the underlying coordinating mechanism remains the same.

(4) **Meeting place suggestions**
Member students can share their explored locations and make suggestions for the whole community to gather. We name this feature 'Wander'. This provides a sense of freedom and satisfaction of acceptance for the contributors. Administrators will have to verify these locations are appropriate and safe. The approved locations are then made available automatically for use by other features in the app. This removes some limitations for student activities, such as tutoring, study groups, or textbook exchanges, now students can meet outside of Hunter at places themselves suggested.

(5) **Honor system**
Agora based its success on member contribution. The better the member society, the better the application becomes. Agora provides a rating system that is not influenced by third-party special interest groups. Members can rate and comment on their experiences for any event they participate in. They can rate organizers for tutoring and study groups anonymously. They can rate their textbook exchanges. Administrators will just be the monitoring party to prevent inappropriate behaviors from occurring.

4 IMPLEMENTATION

(1) **Infrastructure Design**
Our infrastructure relies on the robust connectivity and hosting services available on the public internet (See figure 1 below).

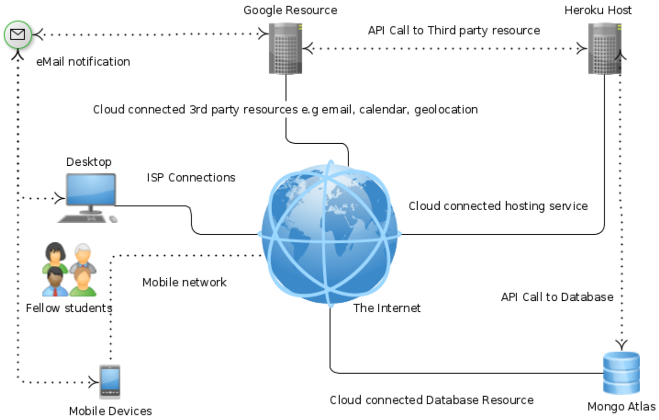

Google Resource
Heroku Host
API Call to Third party resource
eMail notification
Cloud connected 3rd party resources e.g email, calendar, geolocation
Desktop
ISP Connections
Cloud connected hosting service
Fellow students
Mobile network
The Internet
API Call to Database
Mobile Devices
Cloud connected Database Resource
Mongo Atlas

Figure 1: Design infrastructure

(a) **User Environment**
The web site is built using predefined bootstrap styles, we built it with flexible frames so pages can adjust dynamically to user devices capabilities. The majority of the layout was designed for full-size desktops to properly display the vast amount of information available. However, it can adjust to the screen sizes of mobile devices to provide basic functionality such as accepting requests.

(b) **Hardware resource**
We developed the application with a local computing environment and within a local network. The code can easily be ported to be hosted on commercially available resources or can stay local if network and computing capacity is not an issue. We later hosted the website on Heroku, a free web developing resource. Based on the results from performance testing (Fig. 3), we can see that our design is robust and portable. Our database resource is also hosted in the cloud, this frees us from maintaining a local database.

(2) **Front-End**

(a) **Portability**
The application relies on Nodejs to build the underlying server component. We are utilizing bootstrap and Bootswatch[13] schemes as the formatting baseline for our user interface. This gives us the flexibility of adjusting the code for Mobile device view, our current implementation is based on a full-size desktop.

(b) **Functionality**
Given the vast amount of functionalities within Agora, quite a few user interfaces were developed. The complete application takes well over 25 user interfaces to provide just the bare-bone functionality. As the application matures, more elaborate routed pages can provide members with additional features to aid their social life.

(c) **Gadgets**
We incorporated Google calendar in the user dashboard to provide a real-time overview of upcoming events. We provided a news ticker in the same dashboard to remind users not to lose sight of the real world. Leaflet[1], an open-source library for interactive maps, is also incorporated into various pages to plot locations that are saved into the database onto a map. Users are only one click away in exploring event locations when needed. We hope to provide the most convenient experience for students to access all of their resources.

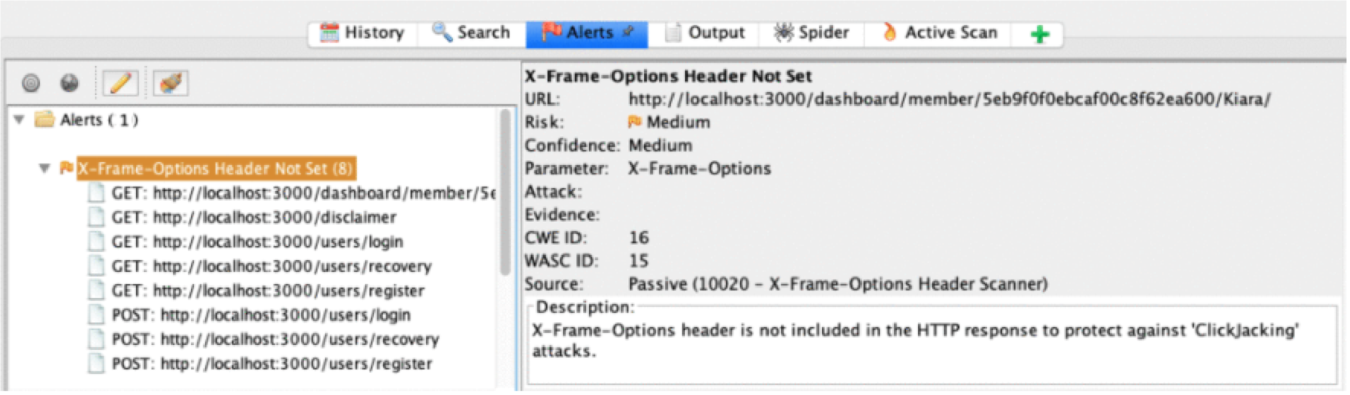
(d) **Security and privacy**
To protect user credentials, their password is encrypted using bcrypt[14] before it was transmitted over the network, and is saved in the database in an encrypted format. Email verification is incorporated to make sure members have a legitimate email address. They are required to respond using their registered email in order to create an account or reset passwords. The website also detects user inactivity. After 15 minutes of user inactivity, the application signs the user out of their sessions, requiring them to have to enter their credentials into the login page again.

(e) **Login**
The client UI presents User with a sign-in/sign up page. They have the option to create a user account or log in as an existing user. When creating an account, the user is prompted to provide their name, password, username, and an email address. The user then receives an email to verify the address. If signing into an existing account, the user then enters their credentials. Their credentials are encrypted on the client-side. The client UI sends the encryption to the server. The database checks for a match. If there is no match, the user is prompted to re-enter credentials. If the user forgets their password, they can click the 'Forgot password?' option located below the login credential fields on the sign-in page. The user is prompted to enter their registered email address and receives a confirmation code via email. Upon entering the code, the user is prompted to reset their password and confirm. The database is updated with the new password. User then receives an email notification for making the change to their account.

(f) **Dashboards**
After login, users are presented with a dashboard that displays a news ticker of current topics. There is also a shared read-only calendar that allows them to see the events happening within the application, such as meeting events and tutor sessions. They can also navigate to other functions of the application, including Exchange (for textbook resources), Gather (for Meetings and Tutor sessions), Wander (for suggested gathering places), and Profile (where they can edit their account details). The administrator can access the Manage dashboard where they can see the status of the application, such as member count, number of requests pending, and such. Administrators can also modify member accounts by unlocking, disable/enabling, or deleting accounts.

(g) **Exchange**
This feature allows a member to search for available textbooks via a robust dynamically updated Data table[10] using any of the attributes, such as ISBN Numbers, Title, and Author. From here, member and proceed to view the offering details and request for it. They can also contribute by offering their own textbooks. A Bookshelf page shows a list of textbooks in their posses ion, either owned, offered, or borrowed/rented. There is also a history page for them to see their previous transactions.

(h) **Gather**
Gather feature allows members to search for study



```
Alerts ( 1 )
  X-Frame-Options Header Not Set (8)
    GET: http://localhost:3000/dashboard/member/5e
    GET: http://localhost:3000/disclaimer
    GET: http://localhost:3000/users/login
    GET: http://localhost:3000/users/recovery
    GET: http://localhost:3000/users/register
    POST: http://localhost:3000/users/login
    POST: http://localhost:3000/users/recovery
    POST: http://localhost:3000/users/register
```

```
X-Frame-Options Header Not Set
URL:        http://localhost:3000/dashboard/member/5eb9f0f0ebcaf00c8f62ea600/Kiara/
Risk:       Medium
Confidence: Medium
Parameter:  X-Frame-Options
Attack:
Evidence:
CWE ID:     16
WASC ID:    15
Source:     Passive (10020 - X-Frame-Options Header Scanner)
Description:
X-Frame-Options header is not included in the HTTP response to protect against 'ClickJacking'
attacks.
```

Figure 2: OSWAP ZAP Security Testing Results

groups and tutoring sessions. From here they can view, join, or leave a session. They can also offer their own session. They can also rate the organizer based on a 5-star system and provide comments as needed.

(i) **Wander**
This feature allows members to view suggested locations for meetings. They rate their experiences, provide comments, or even suggest new locations for meeting places. Suggested locations will have to be approved by an administrator before they are made available for other resources to use.

(3) **Back-end**

(a) **Components**
The back-end portion of the project which deals with the client-server communication is handled by node express, mongoose, nodemailer, node-geocoder, and various other open-source packages from Node Package Manager(NPM)[12].

(b) **Vendor resources**
We are utilizing a free g-mail account for sending notification emails and a for shared calendar. We are also utilizing Leaflet[1] for mapping locations. The majority of the features and function of this application is retrieving and presenting data that are saved in an online MongoDB and using the google calendar resource.

(c) **Portability**
The application is quite lightweight to host. Abiding by the concept of a RESTful API, information is only rendered as needed. This can be easily ported to commercially available hosting platforms.

(d) **Data maintenance**
By design, we incorporate as much automation for data relevancy into the user interface. This way, users are maintaining the database, updates are performed actively by users instead of requiring constant maintenance. Once up and running, there is no need for administrators to maintain, thus providing a real-time updated information system for users.

(4) **Database**

(a) **Design**
The database that is being used for this project is MongoDB. The database is hosted on MongoDB Atlas.

(b) **Schema**
The schema was defined using Mongoose. As this is not a relational database, it is time-consuming to declare the schema (See Figure 1 in appendices). But once the schema was clearly defined, navigating and retrieving information from the database becomes easier and faster. We took advantage of the free version for the project.

5 EVALUATION

To verify the completeness of our application, we focus on testing to evaluate the security and performance of the application.

(1) **Security Testing**
This test would require ensuring that user information is encrypted and that other users can only be authenticated with the information that they provided into the database. This would also involve guaranteeing that users that are not admins cannot have administrative privileges.
A tool that we used for security testing is OWASP ZAP[16]. This tool assisted in penetration testing and vulnerability testing. While running the automatic penetration test on the application, an alert of 'Medium' risk had emerged stating that "X-Frame-Options Header

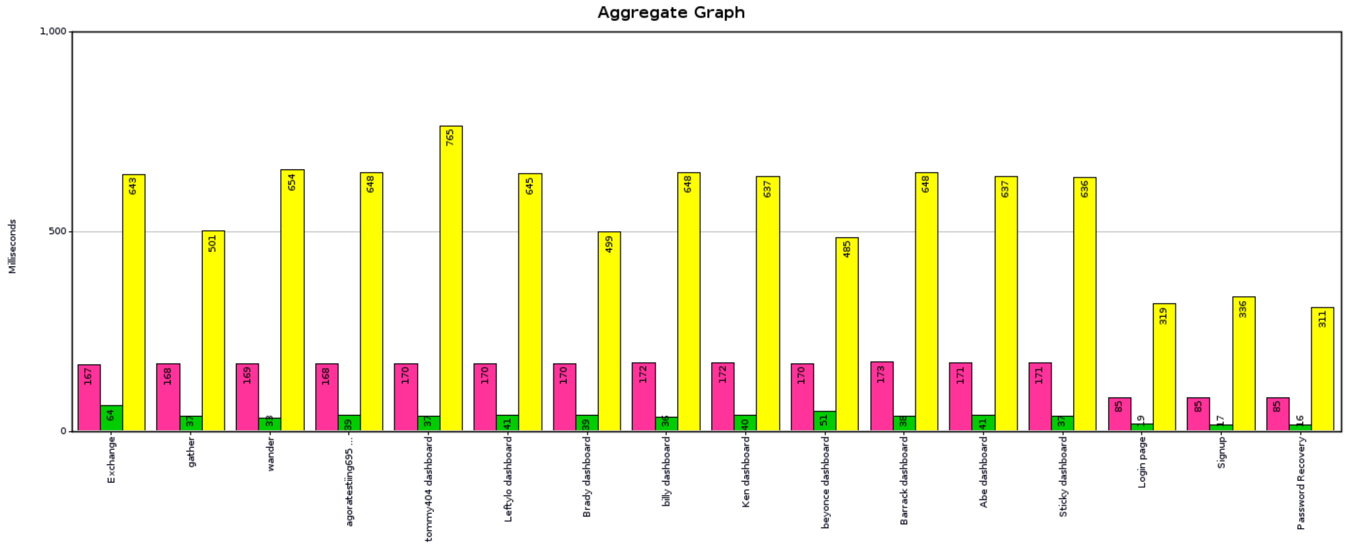


Figure 3: jMeter Response time results

Not Set". In Figure 2, this alert can be seen in the bottom left corner, as well as the details of the alert and potential fixes are displayed in the bottom right corner. Due to these headers not being set, this raised the potential for clickjacking, and possibly stealing information from the user. To fix this, we used the Node package named Helmet[2]. This package is supposed to provide a more secure website by setting various HTTP headers and preventing clickjacking.

(2) **Response Time Testing**
We use Apache JMeter[6] for response time testing. This tool provides load testing when multiple users login to the web server. We use 50 concurrent users accessing various web pages within the application, each ramping up within 5 seconds and repeat for 30 times. We also included 10 concurrent users accessing their dashboard in the test. Average response time for this this is below 200ms when hosted at Heroku. with a minimum of 16ms before user login and a maximum of 765ms after users login and accessing their data.(See Figure 3.)

6 DISCUSSION

(1) **Importance**
Working through developing the project, we realized the importance of user input in achieving the final goal for any community-based application. We, as members of the student community ourselves, understand our needs and difficulties in finding available resources. Therefore we are able to come up with essential functions that we long and wish for. This project also exposes the lack of student life help in our current environment. Approaching this as an assignment initially, we came to realize how much we can do with such an application while putting ourselves in the shoes of member students(which is the easiest part), we continue to ask, 'What else do we need?'. We hope to provide some groundwork for an eventual product that someday will ease the burden of many fellow students.

(2) **Challenges**
We approached this project one step at a time, thanks to the guidance from our instructor. The scheduled due dates for assignments, goals, and check-ins really helped our progress. Gave us a set of goals that mirror real-world project development life cycles. This project also emphasizes our lack of knowledge in production systems. We struggle in learning and incorporating various open-source resources into our project, and learned a lot about the potentials of what we could have done if we are more fluent in those applications.

(3) **Future of Agora**
Given that most of us will conclude our university life soon, we acknowledge the chance for us to continue this application into production is limited. However, we do hope this project does shed some light on the limitations of current student support systems, and maybe the college or someone will take note of our effort and continue the endeavor. Looking at past and various attempts on counterexamples we listed, we came to a

realization that these efforts need constant time and resource contribution, to keep such sites updated and relevant. Any future effort to continue will require a constant sacrifice of student resource, we hope to keep this at a minimum but employing automated mechanism within the app. Regardless of where Agora goes, we believe bits and pieces of this project will definitely make their way into certain future projects we ourselves might face. If not in spirit, at least in pieces.

7 CONCLUSION

This project was conjured up via our understanding of the needs of community college students in our society. We based our design on a 'see a need, fill a need' approach. There are no bells and whistles added, only essential features we hope to integrate. We think this is the most important aspect of any project. It is the heart and soul of any application. With the need in place, there is an abundance of features and functions we can cherry-pick. With the understanding of the need, we are able to pick the best of them (least we thought was the best) and work towards integrating them into the project. Team contribution is also a major factor. No one member is good at doing everything. We were able to split the workload among ourselves with what limited knowledge we had and tried to develop into that role in order to move forward. One main aspect of our realization is in regards to the tools available for us. Yes, we can have the best of all ideas, but without some skill in handling those tools, we are severely handicapped in creation.

8 ACKNOWLEDGES

This project will not be possible if not for open source and free contributors out there. We named some but not all of the resources that helped us piece together the application. Special thanks to all who spent their time sharing their creations. Also not possible if not for our fellow classmates' ideas, and our instructor's guidance. Lastly, the hard work of all team members.

REFERENCES

[1] Vladimir Agafonkin. 2020. *Leaflet.* Retrieved May 12, 2020 from https://leafletjs.com/index.html

[2] Adam Baldwin. 2020. *Helmet-NPM.* Retrieved May 7, 2020 from https://www.npmjs.com/package/helmet

[3] Hunter CUNY. 2020. *Hunter College Community.* Retrieved February 11, 2020 from http://www.hunter.cuny.edu/studentaffairs/student-life/local-community

[4] Hunter CUNY. 2020. *Hunter College Tutoring.* Retrieved February 11, 2020 from http://www.hunter.cuny.edu/advising/my-success-network/learning-centers-tutors

[5] Facebook.com. 2020. *Facebook.* Retrieved February 11, 2020 from https://www.facebook.com/help/572885262883136/?helpref=hc_fnav

[6] Apache Software Foundation. 2020. *Apache JMeter*^{TM}. Retrieved March 25, 2020 from https://jmeter.apache.org/usermanual/get-started.html

[7] CUNY Hunter. 2020. *Hunter College Website.* Retrieved February 11, 2020 from https://hunter.cuny.edu/students/

[8] BriteLand LLC. 2020. *Eventbrite.* Retrieved February 11, 2020 from https://www.eventbrite.com/

[9] Meetup LLC. 2020. *Meetup.* Retrieved February 11, 2020 from https://www.meetup.com/

[10] SpryMedia Ltd. 2020. *Datatables.* Retrieved March 25, 2020 from https://www.datatables.net/

[11] Joshua J. Mark. 2009. *Agora.* Retrieved February 11, 2020 from https://www.ancient.eu/agora/

[12] npm inc. 2020. *NPM.* Retrieved March 25, 2020 from https://www.npmjs.com/

[13] Thomas Park. 2020. *Bootswatch.* Retrieved March 25, 2020 from https://bootswatch.com/superhero/

[14] Johnny Shelley. 2002. *Bcrypt.* Retrieved March 25, 2020 from http://bcrypt.sourceforge.net/

[15] Student2Student.com. 2020. *Student2Student.* Retrieved February 11, 2020 from https://student2student.com/

[16] Zaproxy.org. 2020. *OWASP ZAP.* Retrieved March 25, 2020 from https://www.zaproxy.org/getting-started/

Appendices

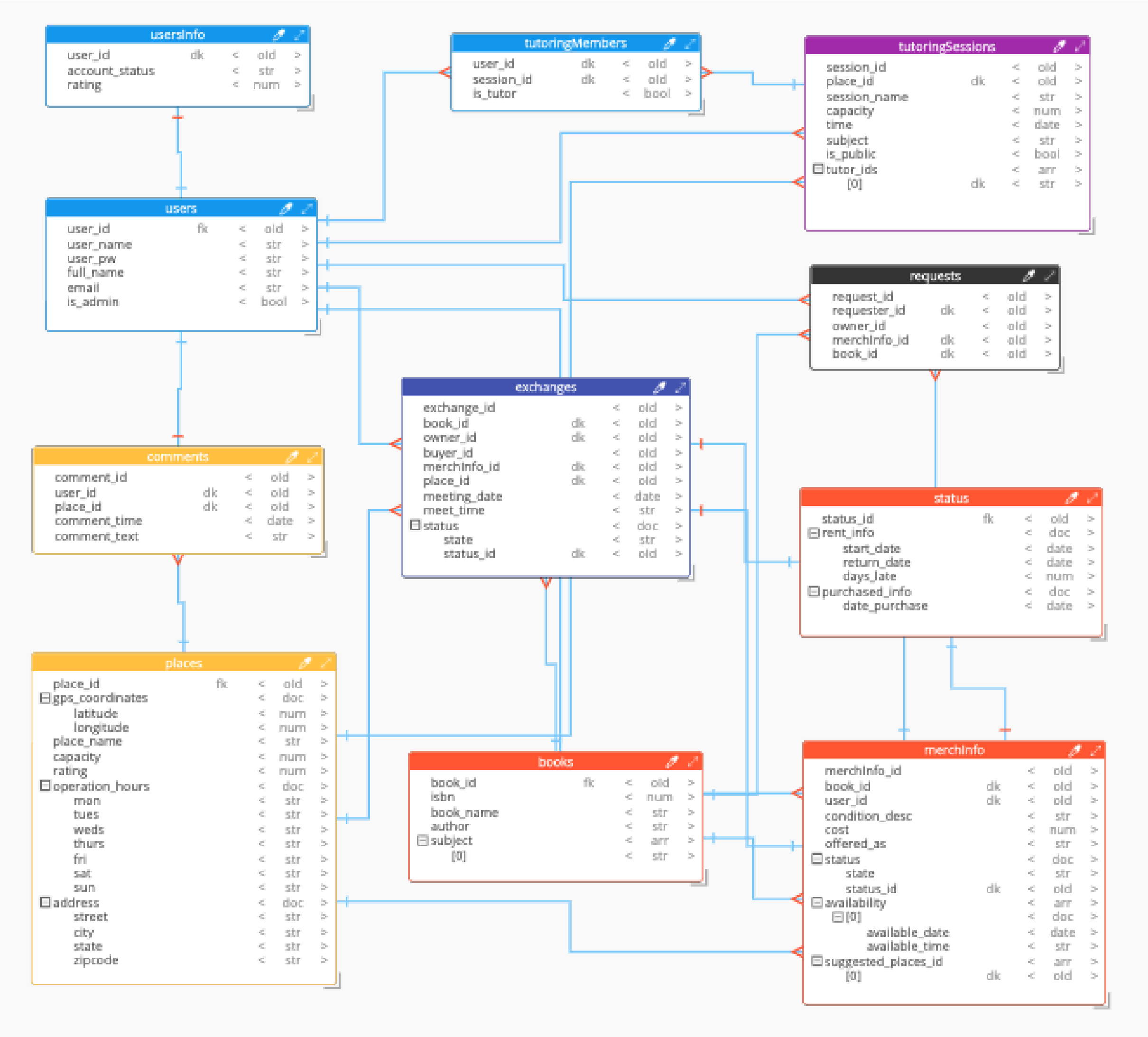
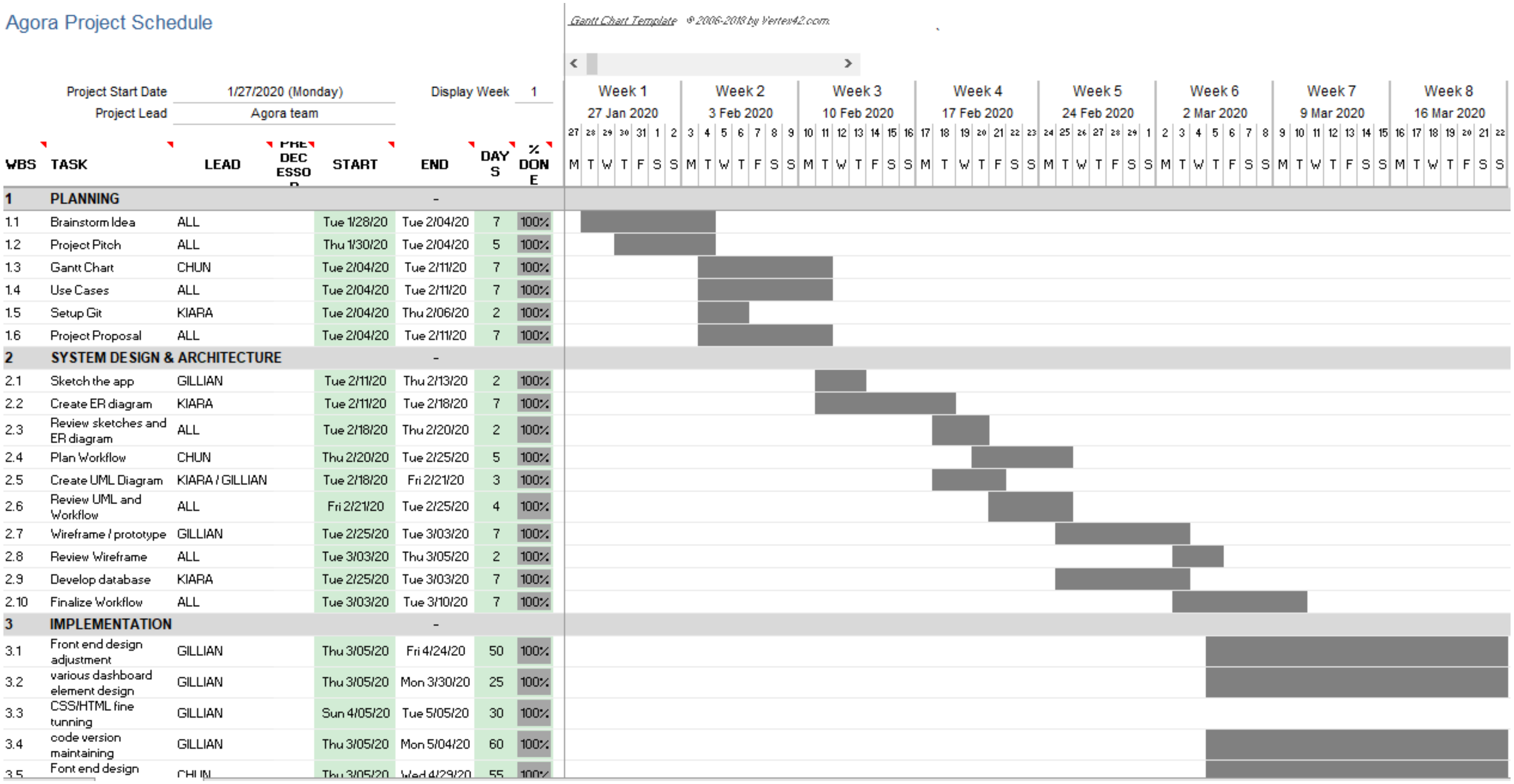


Figure 4: Entity Relationship Diagram



Agora Project Schedule

Project Start Date 1/27/2020 (Monday)

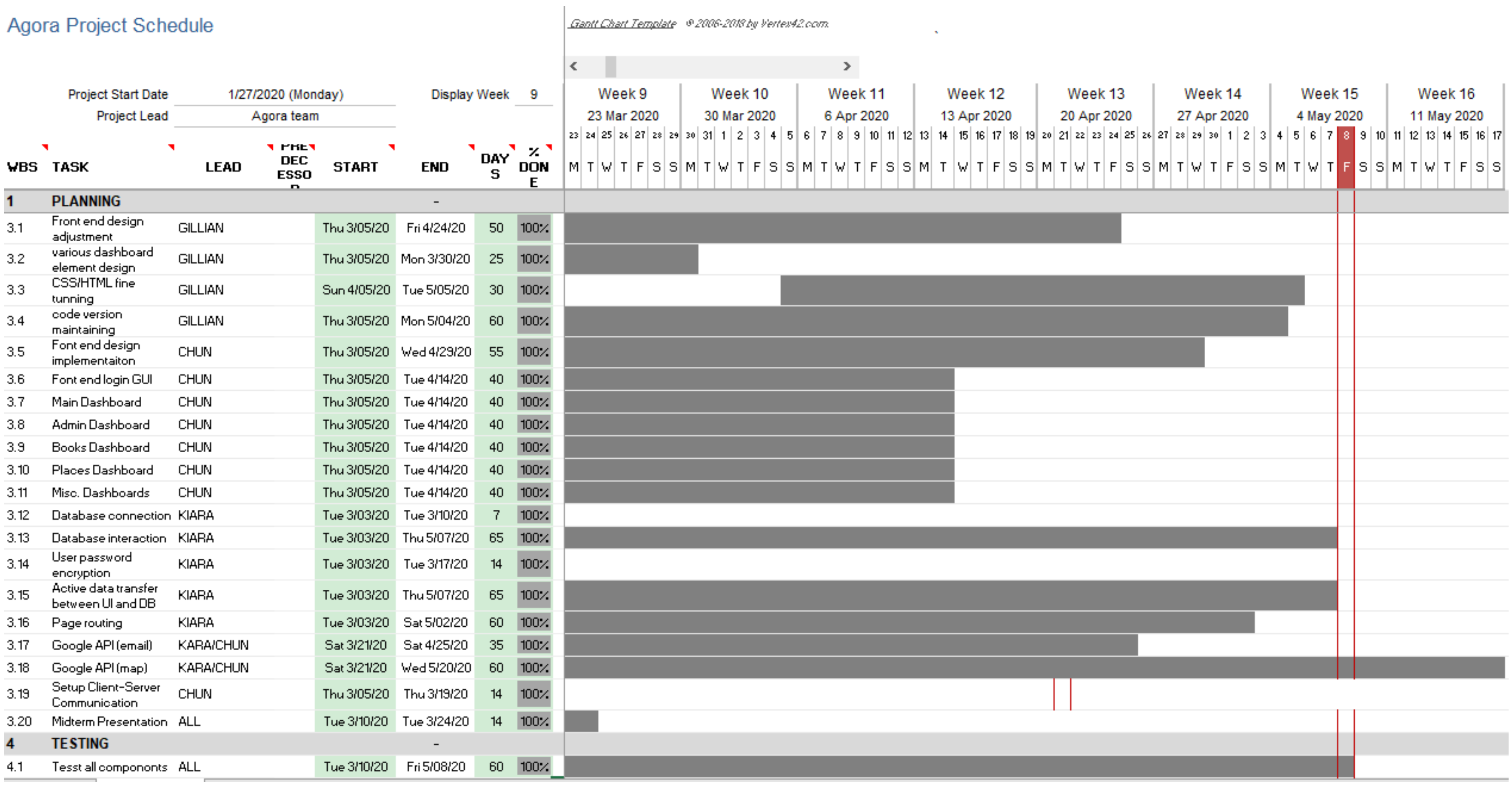
Project Lead Agora team

Display Week 1

Gantt Chart Template © 2006-2018 by Vertex42.com

WBS	TASK	LEAD	PREDECESSO	START	END	DAYS	% DONE
1	PLANNING				-		
1.1	Brainstorm Idea	ALL		Tue 1/28/20	Tue 2/04/20	7	100%
1.2	Project Pitch	ALL		Thu 1/30/20	Tue 2/04/20	5	100%
1.3	Gantt Chart	CHUN		Tue 2/04/20	Tue 2/11/20	7	100%
1.4	Use Cases	ALL		Tue 2/04/20	Tue 2/11/20	7	100%
1.5	Setup Git	KIARA		Tue 2/04/20	Thu 2/06/20	2	100%
1.6	Project Proposal	ALL		Tue 2/04/20	Tue 2/11/20	7	100%
2	SYSTEM DESIGN & ARCHITECTURE				-		
2.1	Sketch the app	GILLIAN		Tue 2/11/20	Thu 2/13/20	2	100%
2.2	Create ER diagram	KIARA		Tue 2/11/20	Tue 2/18/20	7	100%
2.3	Review sketches and ER diagram	ALL		Tue 2/18/20	Thu 2/20/20	2	100%
2.4	Plan Workflow	CHUN		Thu 2/20/20	Tue 2/25/20	5	100%
2.5	Create UML Diagram	KIARA / GILLIAN		Tue 2/18/20	Fri 2/21/20	3	100%
2.6	Review UML and Workflow	ALL		Fri 2/21/20	Tue 2/25/20	4	100%
2.7	Wireframe / prototype	GILLIAN		Tue 2/25/20	Tue 3/03/20	7	100%
2.8	Review Wireframe	ALL		Tue 3/03/20	Thu 3/05/20	2	100%
2.9	Develop database	KIARA		Tue 2/25/20	Tue 3/03/20	7	100%
2.10	Finalize Workflow	ALL		Tue 3/03/20	Tue 3/10/20	7	100%
3	IMPLEMENTATION				-		
3.1	Front end design adjustment	GILLIAN		Thu 3/05/20	Fri 4/24/20	50	100%
3.2	various dashboard element design	GILLIAN		Thu 3/05/20	Mon 3/30/20	25	100%
3.3	CSS/HTML fine tunning	GILLIAN		Sun 4/05/20	Tue 5/05/20	30	100%
3.4	code version maintaining	GILLIAN		Thu 3/05/20	Mon 5/04/20	60	100%
3.5	Font end design	CHUN		Thu 3/05/20	Wed 4/29/20	55	100%

Figure 5: Gantt Chart 1 of 2



Agora Project Schedule

Project Start Date 1/27/2020 (Monday)

Project Lead Agora team

Display Week 9

WBS	TASK	LEAD	PREDECESSOR	START	END	DAYS	% DONE
1	PLANNING				-		
3.1	Front end design adjustment	GILLIAN		Thu 3/05/20	Fri 4/24/20	50	100%
3.2	various dashboard element design	GILLIAN		Thu 3/05/20	Mon 3/30/20	25	100%
3.3	CSS/HTML fine tunning	GILLIAN		Sun 4/05/20	Tue 5/05/20	30	100%
3.4	code version maintaining	GILLIAN		Thu 3/05/20	Mon 5/04/20	60	100%
3.5	Font end design implementaiton	CHUN		Thu 3/05/20	Wed 4/29/20	55	100%
3.6	Font end login GUI	CHUN		Thu 3/05/20	Tue 4/14/20	40	100%
3.7	Main Dashboard	CHUN		Thu 3/05/20	Tue 4/14/20	40	100%
3.8	Admin Dashboard	CHUN		Thu 3/05/20	Tue 4/14/20	40	100%
3.9	Books Dashboard	CHUN		Thu 3/05/20	Tue 4/14/20	40	100%
3.10	Places Dashboard	CHUN		Thu 3/05/20	Tue 4/14/20	40	100%
3.11	Misc. Dashboards	CHUN		Thu 3/05/20	Tue 4/14/20	40	100%
3.12	Database connection	KIARA		Tue 3/03/20	Tue 3/10/20	7	100%
3.13	Database interaction	KIARA		Tue 3/03/20	Thu 5/07/20	65	100%
3.14	User password encryption	KIARA		Tue 3/03/20	Tue 3/17/20	14	100%
3.15	Active data transfer between UI and DB	KIARA		Tue 3/03/20	Thu 5/07/20	65	100%
3.16	Page routing	KIARA		Tue 3/03/20	Sat 5/02/20	60	100%
3.17	Google API (email)	KARA/CHUN		Sat 3/21/20	Sat 4/25/20	35	100%
3.18	Google API (map)	KARA/CHUN		Sat 3/21/20	Wed 5/20/20	60	100%
3.19	Setup Client-Server Communication	CHUN		Thu 3/05/20	Thu 3/19/20	14	100%
3.20	Midterm Presentation	ALL		Tue 3/10/20	Tue 3/24/20	14	100%
4	TESTING				-		
4.1	Tesst all compononts	ALL		Tue 3/10/20	Fri 5/08/20	60	100%

Figure 6: Gantt Chart 2 of 2