BCGL: Barcode Grocery List

Design Document 1

Final Draft v2.0

March 8, 2021

RAID Inc.

RAID INC.

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Rough Draft: Version 1.0	Bailey Martin, Andrew Moore, Josh Davidson, Kyle Hieb	*BCGLDesignDocument1.docx* provides detailed design information on RAID Inc.'s proposed Barcode Grocery List (BCGL).	3/1/21
Final Draft: Version 2.0	Kyle Hieb, Bailey Martin	*BCGLDesignDocument1.docx* provides detailed design information on RAID Inc.'s proposed Barcode Grocery List (BCGL).	3/8/21

Contents

1 Introduction

1.1 Overview

The Barcode Grocery List (BCGL) is a mobile application where users can use their smart devices to scan barcodes (SKU or UPC labels) and look up products. These products can then be added to various shopping lists. For example, a person could make a grocery list and a Christmas list. This app will be available for use by the general public and will be usable on both iOS and Android devices.

The purpose of this document is to provide an overview of the general architectural design plan for the BCGL software. We will present a list of deliverable items and their associated dates of delivery. In addition, this document will contain an overview of the project organization, our team's managerial and technical processes, and a completed risk analysis. In this document, we will review the current project plan, which we will continue to follow to ensure all of our tasks are completed before their due dates.

1.2 List of Deliverables

The following is a list of the release of our deliverables, which includes the dates that they will be submitted by. It is important to note that some of these dates have already passed, and the associated deliverables have already been submitted.

- **January 2021**
  13th: Project Topic Selection Due
  27th: Software Requirements Specification Document Rough Draft Due
- **February 2021**
  3rd: Project Blog/Website must be online
  8th: Software Requirements Specification Document Final Version Due
  10th: Press Release 1 Rough Draft Due, Project Schedule Rough Draft Due
  15th: Press Release 1 Final Version Due
  17th: Project Schedule Final Version Due
  22nd: Introductory Project Presentation
- **March 2021**
  1st: Design Document 1 Rough Draft Due
  8th: Design Document 1 Final Version Due
  22nd: Design Document 2 Rough Draft Due
  29th: Design Document 2 Final Version Due
  31st: Status Meeting/Presentation, Prototypes Due
- **April 2021**
  7th: Press Release 2 Rough Draft Due
  12th: Press Release 2 Final Version Due

14th: User's Manual Rough Draft Due
21st: User's Manual Final Version Due
26th: Project Due
26th: Final Presentations
28th: Final Report Due

1.3 Scheduled Review of Plans

Throughout the duration of the software development project, the RAID Inc. development team will participate in periodic, formal project schedule review sessions. These meetings will help to ensure that the team is staying on schedule and will allow them to take any corrective actions that may be needed. Although the team is constantly making themselves aware of upcoming due dates, these formal checkpoints are designed to further benefit the team. Here are four dates that are distributed across the software development project timeline:

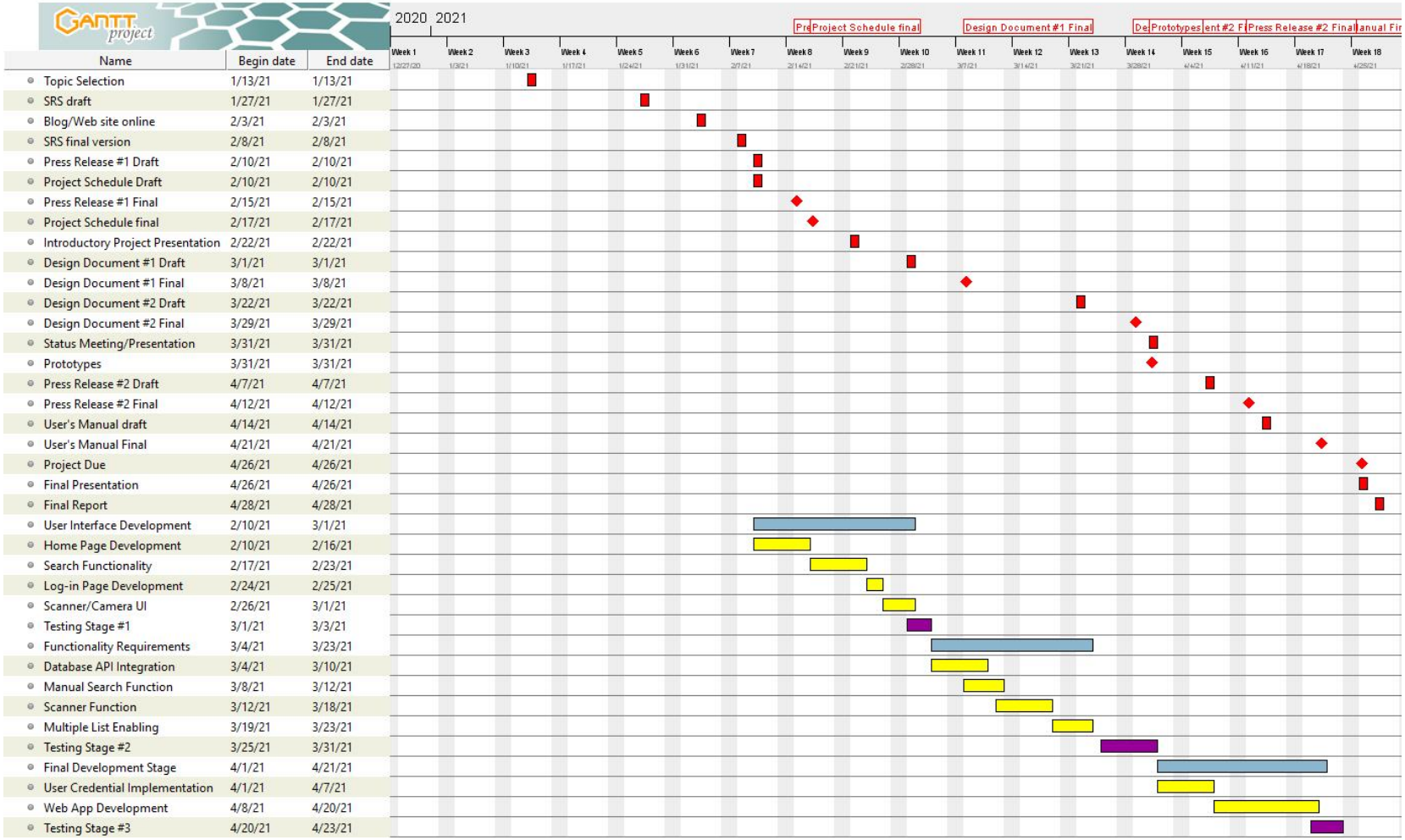
February 22, 2021: Introductory Presentation

March 22, 2021: Design Document 2 Rough Draft Due and Prepare for Prototype Meeting

April 14, 2021: User's Manual Due, 2 weeks before Final Project due date

April 28, 2021: Final Project Due

1.4 Project Schedule



Name	Begin date	End date
Topic Selection	1/13/21	1/13/21
SRS draft	1/27/21	1/27/21
Blog/Web site online	2/3/21	2/3/21
SRS final version	2/8/21	2/8/21
Press Release #1 Draft	2/10/21	2/10/21
Project Schedule Draft	2/10/21	2/10/21
Press Release #1 Final	2/15/21	2/15/21
Project Schedule final	2/17/21	2/17/21
Introductory Project Presentation	2/22/21	2/22/21
Design Document #1 Draft	3/1/21	3/1/21
Design Document #1 Final	3/8/21	3/8/21
Design Document #2 Draft	3/22/21	3/22/21
Design Document #2 Final	3/29/21	3/29/21
Status Meeting/Presentation	3/31/21	3/31/21
Prototypes	3/31/21	3/31/21
Press Release #2 Draft	4/7/21	4/7/21
Press Release #2 Final	4/12/21	4/12/21
User's Manual draft	4/14/21	4/14/21
User's Manual Final	4/21/21	4/21/21
Project Due	4/26/21	4/26/21
Final Presentation	4/26/21	4/26/21
Final Report	4/28/21	4/28/21
User Interface Development	2/10/21	3/1/21
Home Page Development	2/10/21	2/16/21
Search Functionality	2/17/21	2/23/21
Log-in Page Development	2/24/21	2/25/21
Scanner/Camera UI	2/26/21	3/1/21
Testing Stage #1	3/1/21	3/3/21
Functionality Requirements	3/4/21	3/23/21
Database API Integration	3/4/21	3/10/21
Manual Search Function	3/8/21	3/12/21
Scanner Function	3/12/21	3/18/21
Multiple List Enabling	3/19/21	3/23/21
Testing Stage #2	3/25/21	3/31/21
Final Development Stage	4/1/21	4/21/21
User Credential Implementation	4/1/21	4/7/21
Web App Development	4/8/21	4/20/21
Testing Stage #3	4/20/21	4/23/21

1.5 Definition of Terms and Acronyms Used

Agile: Software life-cycle model that stresses the importance of customer communication throughout the entire development process.

API: *Application Programming Interface*. An API is used to provide a resource of data and functionality to a software development project. Developers can utilize calls and requests to creatively implement the data that the API contains into their specific project.

BCGL: *Bar Code Grocery List*. This is the product that is being created.

Git/GitHub: Source code management for storing and viewing developed code.

Kanban: Japanese production task scheduling system that often utilizes a Kanban board to organize project tasks.

MVC Framework: *Model View Controller*. This software development approach segments the code for the application's model, view, and controller into three distinct partitions. This helps developers stay organized and allows for universal readability.

RAID Inc.: Company developing the BCGL software.

SKU: *Stock Keeping Unit*. This is the number on a barcode, which is unique to each product.

SLCM: *Software Life Cycle Model*. A term used in the software engineering community to describe the approach utilized in a software development project.

UPC: *Universal Product Code*. This is the number displayed on a barcode, which is unique to each product.

Xamarin: Cross-platform mobile application development infrastructure.

2 Project Organization

2.1 Software Life Cycle Model

The Kanban Agile method will be utilized as the Software Life Cycle Model for this project. However, instead of conducting a daily stand-up, we will perform a stand-up at every meeting. We will have weekly stand-ups as our worst-case scenario. However, they will likely take place more frequently, as we will often hold meetings several times per week. The Kanban approach is an appropriate choice for this software development project, as it emphasizes the importance of communication. This is especially important as this is a team project, and each team member will often be working on different tasks and components of the project. We will also utilize a portion of the Design-to-Schedule Software Life Cycle Model, as we will prioritize our requirements and deliver a final product by the due date of the project (April 26, 2021).

As we are using the Kanban approach, we need to utilize a Kanban Board to accompany it. The Kanban board contains cards that outline the tasks that we have to begin working on, those we are currently working on, items we are in the process of testing, and work items that have been completely finished. Task cards are moved across the board columns as the state of the specified task changes.

2.1.1 Justification of SLCM

The reason for choosing Kanban is because of the board design and its associated features. The primary reason for wanting to utilize a Kanban board is that it helps to keep teams organized. We will be able to look at it and easily identify which tasks need to be completed, and in what order. The prioritization of tasks based on their column placement on the board makes it visually helpful. This allows us to immediately determine the overall status of the project. Also, being able to manually "pull" items through the Kanban development process will help us visually follow the entire software engineering process. A standard Kanban board features three columns. The columns are traditionally named To-Do, In-Progress, and Completed. This will help our group visually stay grounded and focused from an organizational perspective. Furthermore, the board possesses a work-in-progress limitation on the number of cards or tasks that we can have in a column at any time. This limit will help our team from becoming overwhelmed and eliminates potential bottlenecks. This work-in-progress limit forces us to successfully complete assigned tasks before moving on and attempting other tasks.

We chose the agile approach because it focuses mainly on customer and user experience, instead of the developer experience. We are not designing this product for the developers' satisfaction; we are designing it for the client and project stakeholders. The agile method reflects this philosophy. Also, agile projects are reactive, rather than predictive. This is important because we cannot predict the changes that will be made to the external barcode database in the future. Therefore, the program must be dynamic and react, or adapt, to those changes as they are made.

2.2 Team Member Responsibilities

Bailey Martin is the team leader, meaning that he will be responsible for scheduling the team meetings, keeping the team motivated and on schedule, and for ensuring that project operations run smoothly during stages of deliverable preparation.

As the configuration management specialist, Josh Davidson is responsible for making sure that the completed deliverables are submitted by their assigned due dates. He is in charge of physically delivering the items and for keeping track of the team's progress.

Andrew Moore is the project architect. Andrew's primary role is to develop the BCGL application and to ensure that it runs as intended. He will also be in charge of deploying the database API and for assisting with the submission of the app to both the Apple and Android app stores.

Kyle Hieb serves as the lead tester and is responsible for testing all aspects of the app. The testing process will occur throughout the development effort as well as upon completion of the project. He ensures that the entire program runs smoothly and as originally planned.

Despite these specific roles and job responsibilities, all team members will collaborate with each other on all aspects of development, as captained by each person whose role fits the task at hand. All team members are responsible for each deliverable as well as for the final BCGL software product.

3 Managerial Process

3.1 Team Reporting and Monitoring Mechanisms

Various forms of team reporting and project monitoring mechanisms will be utilized throughout the BCGL software development project. RAID Inc.'s primary method of team reporting and project monitoring will occur via team meetings. We have scheduled weekly meetings to discuss and work on tasks that need to be completed before the next meeting. In addition to these routinely scheduled meetings, we usually have at least one impromptu meeting per week. These additional informal meetings are beneficial, as they allow us to verify that we are all on the same page in regard to the work that we have completed. In addition, these meetings allow us to discuss obstacles that we have individually encountered and give us the additional opportunity to collaborate on project tasks as a group.

These meetings are also designed for the purpose of project scheduling. The team leader is able to provide an update of the project timeline. Additionally, the team leader's updates help the team to ensure they are not falling behind. We also discuss deliverables and making sure that we are all aware of the current state of the project. In doing these things, we are able to monitor our progress, ensuring that we are on schedule.

Our Gantt Chart also serves as another monitoring mechanism. It serves as a visual timeline of all tasks that are associated with all aspects of the project. This is helpful, as all team members are able to see when individual components are due, as well as dependencies that tasks have on each other. As mentioned previously, we will also be utilizing Kanban boards. RAID Inc.'s GitHub Kanban board contains tasks pertaining to deliverables and other project due dates. We also created a second Kanban board on Azure DevOps, which houses our development tasks. Separating these tasks across two boards helps to keep us organized and to easily distinguish deliverable due dates from development due dates. We have identified each functional requirement as high, medium, or low priority. Additionally, the vertical position of a task card in a column on a Kanban board indicates its priority. Tasks higher in a given column have a higher priority and should be worked on first, compared to those located underneath several tasks in a column.

Furthermore, the team leader's role is designed to aid in reporting and the monitoring of team progress. Bailey has also been using Clockify, a collaborative time-keeping software, to log team time and to generate reports about progress and work on specific project items. He opens team meetings by discussing the upcoming week's plans and goals. Lastly, the RAID Inc. team blog (https://bailey-martin.github.io/CSC492-BCGL/) maintains a log of accomplished items and provides another method for documenting project progress and team accomplishments.

4 Technical Process

4.1 Tools

In creating this project, we will be developing our code in Visual Studio 2019. We will be using Xamarin, which enables a developer to write code for the application in C# and then deploy the app on both iOS and Android. The advantage of Xamarin is that developers can write universal code rather than creating individual code for each platform. This is extremely beneficial because it gives us a shorter window for the amount of time needed for development, as it greatly diminishes the amount of code that needs to be unique. Additionally, a Model-View-Controller (MVC) approach will be used in the code design. This will help to keep all source code organized, and to increase the overall level of code readability. Our progress will be managed using GitHub as the chosen Configuration Management software. GitHub allows all members of the team to have access to all parts of the development that other members might have worked on. GitHub also provides our team with version control capabilities. It also offers other collaborative features including the use of development branches and merging abilities.

4.2 Documentation Strategy

All tasks will be tracked on one of our two Kanban boards, which allow us to quickly determine what has been completed, what we are currently working on, and what we still need to begin. One of these boards is stored in GitHub, which tracks the deliverables that we have to submit, ordered by their due dates. The other board is in Microsoft Azure DevOps and tracks the tasks that we still have to do regarding the actual development of the app, as well as all testing that will be done. Both Kanban Boards will be updated as the state of a given task changes.

We will also be documenting weekly summaries, development processes, and all deliverables on our project blog site. This site is hosted by GitHub using GitHub Pages. Deliverables themselves also serve as a form of project progress documentation. This website provides our team, as well as the general public, an avenue to keep track of our project progress thus far.

5 Risk Analysis

5.1 Risk Identification

Risk is inevitable, as it is involved in every project and impacts our daily lives. RAID Inc. worked to identify potential risks associated with the BCGL software development project. These risks have been placed in a bulleted list below.

- Low level of familiarity with our development environment (Xamarin)
- Juggle of other classes are challenging for the team members
- Key piece of project depends on one portion of the project (the barcode API)
    - Delays in this vital piece will affect the project greatly
- Information that we need the API to contain may not be in the API
    - Information could also be outdated
- Various formats of barcodes
- Unable to scan/read barcode
- Users manually searching the database for items that do not exist in the database
- Update doesn't update itself

After compiling a list of potential risks to the project, RAID Inc. performed a risk analysis. In conducting this risk analysis, the likelihood, level of impact, and the overall risk severity were calculated and listed underneath each potential project risk.

- Low level of familiarity with our development environment (Xamarin)
    - Likelihood: high
    - Impact: low
    - Overall risk severity: medium
- Juggle of other classes are challenging for the team members
    - Likelihood: medium
    - Impact: medium
    - Overall risk severity: medium
- Key piece of project depends on one portion of the project (the barcode API)
    - Delays in this vital piece will affect the project greatly
    - Likelihood: high
    - Impact: medium
    - Overall risk severity: high
- Information that we need the API to contain may not be in the API
    - Information could also be outdated
    - Likelihood: low
    - Impact: medium
    - Overall risk severity: low
- Various formats of barcodes
    - Likelihood: low
    - Impact: low

    - Overall risk severity: none
- Unable to scan/read barcode
    - Likelihood: medium
    - Impact: low
    - Overall risk severity: low
- Users manually searching the database for items that do not exist in the database
    - Likelihood: medium
    - Impact: low
    - Overall risk severity: low
- Database doesn't update itself
    - Likelihood: medium
    - Impact: medium
    - Overall risk severity: medium

5.2 Risk Management Strategies

- Low level of familiarity with our development environment (Xamarin)
    - Overall risk severity: medium
    - Response: allocate time learning the environment, make demo/sample programs to get used to it, review documentation
- Juggle of other classes are challenging for the team members
    - Overall risk severity: medium
    - Response: stick to scheduled team meeting times, continue to prioritize the project as a high priority
- Key piece of project depends on one portion of the project (the barcode API)
    - Delays in this vital piece will affect the project greatly
    - Overall risk severity: high
    - Response: make this a priority early in the development process, make beta versions of each phase before moving on, build a solid foundation for the project with this
- Information that we need the API to contain may not be in the API
    - Information could also be outdated
    - Overall risk severity: low
    - Response: research the API's contained information, look for additional resources, manually supplement information where needed, research APIs and choose the one with the best offerings
- Various formats of barcodes
    - Overall risk severity: none
    - Response: providing a manual search bar that allows for manual search functionality (serves as a workaround)
- Unable to scan/read barcode
    - Overall risk severity: low

    - Response: providing a manual search bar that allows for manual search functionality (serves as a workaround)
- Users manually searching the database for items that do not exist in the database
    - Overall risk severity: low
    - Response: have a system in place to catch this event and notify us so that we can add it or find an API with more item offerings
- Database doesn't update itself
    - Overall risk severity: medium
    - Response: reevaluate our choice of API and possibly choose one that does offer an update functionality

6 Architectural Design Specification (DFD)