Software Development Capstone

Proposal

Project Name: Academic Organizer

Author: Nelson C. Araujo

Date: July 22nd, 2022

Version: 1.0

Table of Contents

Executive summary

This proposal details the planning, design, testing, and documentation of an academic planning and tracking Android application for Better Yourself LLC. The proposed application will be designed from a student's perspective and will allow the student to organize each school year's term, courses per term, and track assessments per course. This application will provide the student the ability to organize and plan their academic program from start to graduation. In addition, it will provide a high-level view of when terms and courses start and end as well as when assessments should be started and their due date.

Business improvement opportunity

Better Yourself LLC has a long history of providing excellent education and learning opportunities to thousands of students but currently lacks a way for the students to plan and track their academic progress and provide a reminder of when assessments are due. This application will provide students the ability to organize their academic program and remind the student when assessments are due to ensure the successful completion of each term and course.

Fulfillment

With the completion and deployment of this application to students, they will have the ability to organize and track their academic year and program. This application will give students an easy-to-view and access location to see what they have completed and more importantly what courses and assessments are coming up. Knowing what courses and assessments are coming up allows the student to know where to focus to complete assessments and courses within the allotted time frame.

In addition to benefiting the students, this application will also assist the staff by allowing staff to review the student's progress, make recommendations to improve their program progress and check with the student if they need any assistance with the coming assessments.

Scope

The primary goal of this project is to provide students with an easy-to-use academic tracker. To reach this goal:

- The application will be created using the Android ecosystem and must support Android version 12 and above.
- SQLite will be used to store data locally on the device.
- The following screen layouts will be created with the specified requirements:

- Home Screen
  - Display the application name.
  - Display and link to a list of terms.
  - Display and link to a list of courses.
  - Display and link to a list of assessments.
  - Display a global search bar that allows the student to search for terms, courses, and assessments.

- List of search results
  - Display and open relevant terms, courses, and/or assessments.

- List of terms
  - Display and open terms.
  - Create a new term.

- List of courses
  - Display and open courses.
  - Create a new course.

- List of assessments
  - Display and open assessments.
  - Display the due date of the assessment.
  - Create a new assessment.

- Terms
  - Display the term title. This field will be user-editable.
  - Display the start and end dates. This field will be user-editable.
  - Display, add, and remove courses.
  - Delete the term. A term must not have any courses assigned to be deleted.

- Courses
  - Display the course title. This field will be user-editable.
  - Display the associated term. This field will be user-editable.
  - Display the start and end dates. This field will be user-editable.
  - Display a course note that can be shared via email and/or SMS. This field will be user-editable.
  - Display the status of the course. The status selection options will be In Progress, Completed, Dropped, and Plan to take. This field will be user-editable.
  - Display the course instructor’s name, phone number, and e-mail address. This field will be user-editable.
  - Display, add, and remove assessments.

        - Display a system alert when start and end dates are reached. The application will not need to be running.
        - Delete the course.

    - Assessments
        - Display the assessment title. This field will be user-editable.
        - Display the associated course. This field will be user-editable.
        - Display the start and due dates. This field will be user-editable.
        - Display the assessment content. This field will be user-editable.
        - System alerts on start and end dates are reached. The application will not need to be running.
        - Delete the assessment.

Software development methodology

The waterfall software development model will be used for this project. The waterfall methodology is best suited for this project because the scope has been clearly defined before the start of the project. This approach will allow for a quick start to the design phase of the project and provide a working alpha version of the software earlier.

In addition, to be able to provide a working alpha version of the software earlier there are many advantages to the waterfall model some of those include:

1. It is easy to understand as it has clearly defined phases.

2. It has been in use for an extended period and most users have some experience with the process.

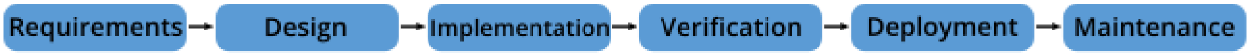
3. Each stage has clear and defined milestones.

4. Requirements and scope are determined early in the project and do not change.

5. Easier to manage as each stage has specific deliverables and a review process.

6. The project process is well documented.

The key phases of the waterfall model are requirements, design, implementation, verification, deployment, and maintenance. As each phase is completed it will progress the project to key milestones and provide key deliverables.



Requirements

All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.

Deliverables

1. Requirements document

Design

The requirement specifications from the first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.

Deliverables

1. Project design document
2. Wireframes
3. Storyboard
4. Mock-ups
5. Class UML
6. Database UML

Implementation

With inputs from the system design, the system is first developed in small programs called units, which are integrated into the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.

Deliverables

1. Functional application
2. Unit test report

Verification

With the application completed internal testing of the application can commence verifying all functionality is present and functional. In addition to developer testing, user acceptance testing is also performed to provide the customer the opportunity to confirm all required functionality is present.

Deliverables

1. User acceptance test report

Deployment

Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.

Deliverables

1. Finalized documentation

Maintenance

Some issues come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

Deliverables

1. Application performance report
2. Application functionality report

Deliverables

The following describes the deliverables that this project will generate through its development process.

Project Deliverables

1. The requirements document will outline the customer requirements and technical requirements related to the project.

2. The project timeline will detail the start and end dates for each milestone in the project.

3. The user acceptance test report will detail the outcome of the user testing. This report will note requirements not met by the product and where improvements are required.

Product Deliverables

1. The product design document will detail how the application is to be developed.

2. Wireframes will provide a low-fidelity view of the application screens.

3. The storyboard will provide a visual workflow of how the different screens in the product will interact.

4. The mock-ups will provide a high-fidelity view of the application screens.

5. The class UML diagram will provide a visual representation of the product’s class structure and methods.

6. The database UML diagram will provide a visual representation of the product’s database structure displaying each table and its fields.

7. A functional database will provide the customer the opportunity to run queries and review the database structure.
8. The screen layouts will provide the actual in-application screens the final product will contain.
9. A functional application will provide the customer the opportunity to evaluate the application.
10. The unit test report will detail the developer's product testing outcome.
11. The product documentation will consist of all the documents created during the development cycle.

Implementation

This application will be developed over 10 weeks and will use the Android platform version 12 and above. After completion of the application, a copy of the fully functional application will be provided to the customer for user acceptance testing. This version of the application may be shared with internal user bases and pre-approved external partner teams. On the acceptance of the application, version 1.0 will be deployed to Google Play for general user usage.

After the completion of this project and implementation of its application, it will provide students the ability to organize their academic program and remind the student when assessments are due to ensure successful completion of each term and course.

Validation and Verification

The application validation and verification process will consist of three phases; unit testing, system testing, and acceptance testing. The first two phases will be completed by the development team in conjunction with the testing team. Phase three will be completed by Better Yourself LLC. The encompassment of these three phases allows for both the developers and Better Yourself LLC. to ensure the application meets all established requirements and is in an acceptable running state.

Unit testing is to be performed as new features are implemented into the application. During this testing phase, the developers and testing team will perform extensive testing of new features. This early testing of features allows bugs to be identified and resolved early in the development phase.

System testing is to be performed when the application is completed by the development team and functions as a final test to ensure the application works as a whole. This testing phase is to be headed by the testing team in coordination with the development team. Specific test cases will be generated based on previous unit tests, the design document, and the requirements document. At a minimum, one test case will be used to verify each of the documented requirements is met.

Lastly, the user acceptance test phase is to be completed by Better Yourself LLC to confirm all requirements are met. This phase provides the internal user base and pre-approved external partner teams access to the application to confirm all required functionality is present and provide feedback for future improvements. Improvement requests not in scope will be documents for future releases.

Environments and Costs

The following outlines the environments and costs associated with the development of this application.

Programming Environment

This application will be developed using Android Studio utilizing the built-in emulator to test features and functionality. The initial development cost is minimal as no additional hardware is required. During the system testing phase, at minimum, one Android 12 phone and tablet is required to perform real-world user testing.

Environment Costs

This application does not incorporate ads and will be a free-to-use application on Google Play, no environmental costs will accrue.

Human Resource Requirements

The development team will account for 41% of the labor hours of this project with project management accounting for 31% and the testing team 30%. The below table breaks down the planned hours per team and phase.

Phase	Development team planned hours	Testing team planned hours	Project management planned hours	Better Yourself planned hours
Requirements			30 hours	
Design	30 hours	15 hours	15 hours	
Implementation	80 hours	80 hours	40 hours	
Verification			5 hours	15 hours
Deployment	30 hours		15 hours	15 hours
Maintenance				
Total:	**140 hours (41%)**	**95 hours (30%)**	**105 hours (31%)**	**30 hours**

Project timeline

The following table outlines the timeline for this project.

Legend

PM: Project Manager
Dev: Development team
Test: Testing Team
Cust: Customer

Phase	Milestone	Deliverable	Dependency	Start	End	Days	Resource
1	Gather the requirements.	Requirements document		05/03	05/08	5	PM
	Determine tools and libraries to be used.		Requirements document	05/08	05/08	1	PM Dev
2	Identify the screen layouts that are needed.		Requirements document	05/09	05/09	0.5	PM Dev
	Identify the requirements for each screen layout.		Requirements document	05/10	05/10	0.5	PM Dev
	Create wireframes for each screen layout.	Wireframes	Requirements document	05/11	05/11	0.5	PM Dev
	Confirm all screen wireframes comply with the requirements.		Requirements document	05/11	05/11	0.5	PM Dev
	Create the storyboard using the wireframes.	Storyboard	Requirements document	05/12	05/12	0.5	PM Dev
	Confirm the storyboard complies with the requirements.		Requirements document	05/12	05/12	0.5	PM Dev
	Create mock-ups for each screen layout.	Mock-ups	Requirements document	05/13	05/13	0.5	PM Dev
	Confirm mock-ups conform to the requirements.		Requirements document	05/13	05/13	0.5	PM Dev
	Create the application UML class diagram.	Class UML	Requirements document	05/14	05/14	1	PM Dev
	Create the database entity relationship diagram(ERD).	Database UML Product design document	Requirements document	05/15	05/15	1	PM Dev
3	Create the database.	Functional database	Product design document	05/16	05/16	0.5	PM Dev
	Create the screen layouts in Android Studio.	Screen layouts	Product design document	05/16	05/16	0.5	PM Dev
	Code and enable the assessment screens functionality	Functional assessment screens	Product design document	05/17	05/29	14	PM Dev
	Code and enable the course screens	Functional course	Product design	05/29	06/05	14	PM

	functionality.	screens	document				Dev
	Code and enable the term screens functionality.	Functional term screens	Product design document	06/05	06/20	14	PM Dev
	Code and enable the home functionality.	Functional application	Product design document	06/21	06/29	8	PM Dev
	Test the assessment screens functionality.		Product design document	06/29	06/29	1	PM Dev Test
	Test the course screens functionality.		Product design document	06/30	06/30	1	PM Dev Test
	Test the term screens functionality.		Product design document	07/01	07/01	1	PM Dev Test
	Test the home screen functionality.	Unit test report	Product design document	07/02	07/02	1	PM Dev Test
4	User acceptance testing	User acceptance test report	Unit test report	07/03	07/10	7	PM Cust
5	Finalize documentation	Finalized documentation	Product design document Unit test report	07/11	07/16	5	PM Dev Test
	Deploy application for general usage.			07/17	07/17	1	PM Cust
6	Monitor application usage and provide bug fixes.	Performance report Functionality report					PM Cust

Gantt chart

The below shows a high-level overview of the project timeline per project phase.

Week	Requirements	Design	Implementation	Verification	Deployment	Maintenance
Week 01 2022 May-03 - May-08	▨					
Week 02 2022 May-09 - May-15		▨				
Week 03 2022 May-16 to May-22		▨				
Week 04 2022 May-23 to May-29			▨			
Week 05 2022 May-30 to June-05			▨			
Week 06 2022 June-06 to June-12			▨			
Week 07 2022 June-13 to June-19			▨			
Week 08			▨			

2022 June-20 to June-26			▨			
Week 09 2022 June-27 to July-03			▨			
Week 10 2022 July-04 to July-10			▨			
Week 11 2022 July-11 to July-17					▨	

Sources

SDLC - Waterfall Model. (n.d). Tutorialspoint.com. https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model/